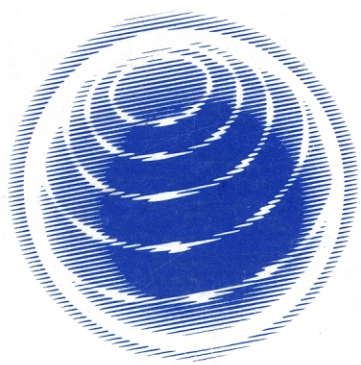


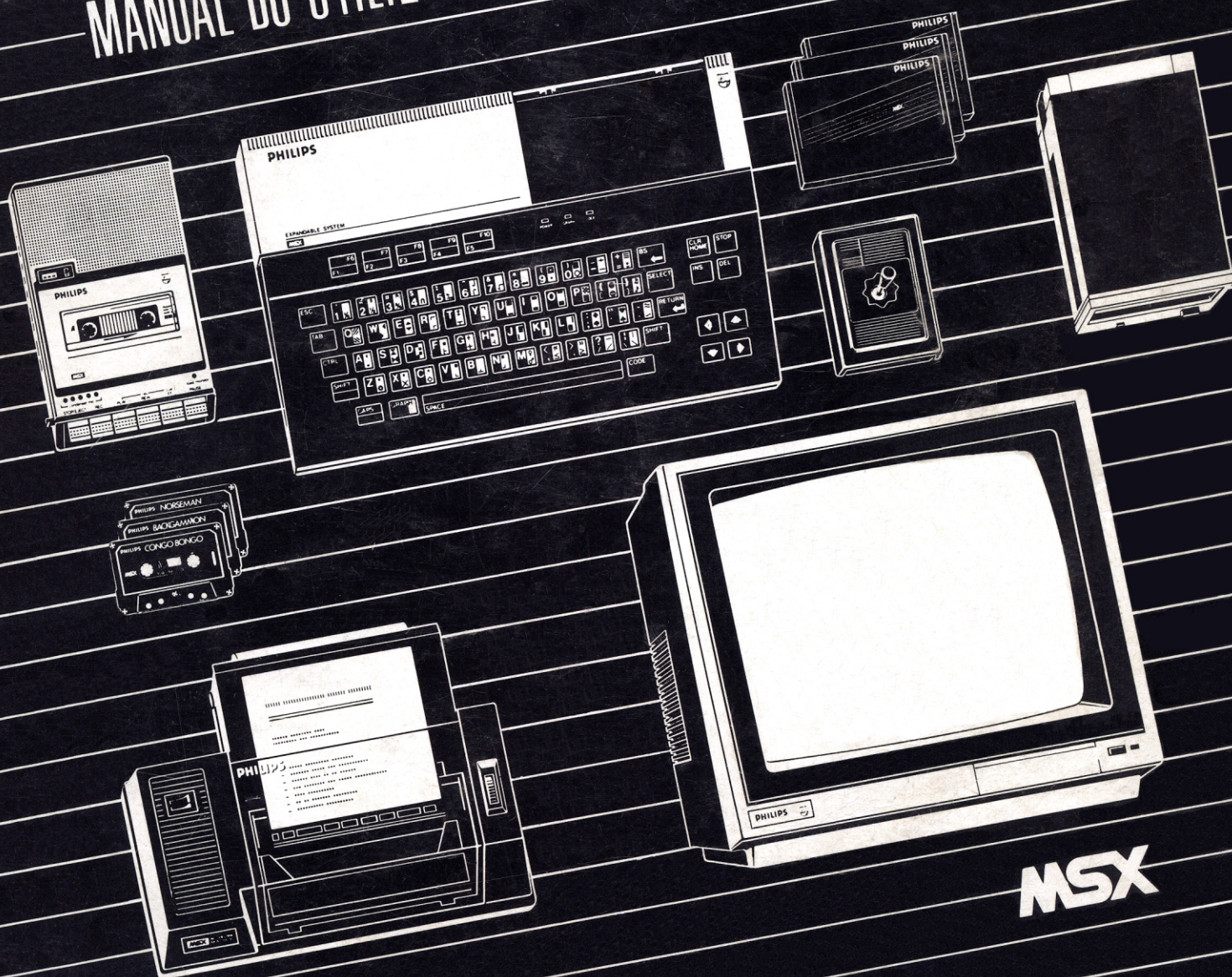


PHILIPS



PHILIPS MSX HOME COMPUTER SYSTEM

MANUAL DO UTILIZADOR



MSX

Philips VG8010

Manual do Utilizador

O presente manual foi digitalizado e restaurado por AKTIMADONNAK PRESS 2013 em Portugal, a partir do livro original, e o objectivo deste é dar a conhecer aos saudosistas do sistema MSX a existência do Manual do Utilizador do computador da Philips, modelo VG8010, que foi comercializado em Portugal nos anos 80.

Como o objectivo é partilhar este magnífico manual que não é mais produzido em nenhuma editora e tão pouco comercializado, é expressamente proibido vender sob qualquer forma conhecida de comércio ou obter qualquer lucro monetário, partes ou o manual completo.

Apenas fiz este trabalho por ter vontade de partilhar este item raro nos dias de hoje, e como tal sem qualquer intenção de venda ou lucro, e para que seja mais um elemento a adicionar às bibliotecas dos fans do mundo MSX.

Este manual é publicado pela Philips Export B. V., Corporate Centre Video, Eindhoven, Holand.

© Philips Export B. V.

Todos os direitos são reservados. A reprodução deste manual, na totalidade ou em partes, por qualquer processo, é proibida sem uma autorização expressa e escrita do editor.

© Microsoft ASCII Corporation

Índice

INTRODUÇÃO

Capítulo 1 INSTALAÇÃO DO VG8010

- 1.1 Desembalar
- 1.2 Fazer as ligações
- 1.3 O teclado

Capítulo 2 MSX-BASIC

- 2.1 Como trabalhar com o MSX-BASIC
- 2.2 Como alterar um programa
- 2.3 Constantes e variáveis
- 2.4 Cálculos
- 2.5 MSX-BASIC e o gravador de dados
- 2.6 Tabelas
- 2.7 Decisões
- 2.8 Sub-rotinas
- 2.9 Funções
- 2.10 MSX-BASIC e o écran
- 2.11 Instruções para as cores e para os gráficos
- 2.12 "Sprites" (movimentação de figuras)
- 2.13 Música
- 2.14 MSX-BASIC e os comandos manuais "joy sticks"

Capítulo 3 MSX HARDWARE

- 3.1 Manutenção
- 3.2 Periféricos

Capítulo 4 APÊNDICES

- A Listagem de erros

B	Funções matemáticas
C	Tabelas de cor
D	Funções de comando — controlo
E	Códigos de caracteres
F	Palavras reservadas
G	Teclado
H	Especificações técnicas
I	Processamento em video
J	Gerador de som programável

AKTIMADONNAK PRESS 2013

Introdução

O VG8010 é um moderno e versátil computador doméstico que pode ser usado para escrever e para aprender os vossos próprios programas do mais simples ao mais complexo. Também pode ser usado numa enorme variedade de programas já prontos.

O VG8010 reúne os últimos requisitos em computadores domésticos: o standard MSX. Oferece-vos uma importante vantagem, já que podeis utilizar qualquer um dos muitos programas da extensa biblioteca MSX. Significa também que tereis enormes possibilidades de escolha em equipamento periférico, tal como impressoras, leitores de discos (disk drives), etc. que podem ser usados em combinação com o vosso VG8010.

O VG8010 tem incorporado um intérprete MSX-BASIC, compatível com o de todos os outros computadores domésticos MSX. O MSX-BASIC reconhece quase todas as instruções da versão Microsoft BASIC80, em complemento com instruções para música, cor, figuras móveis (sprites) e o uso das alavancas de comando (joy sticks).

Neste manual encontrareis em primeiro lugar as instruções para instalar o vosso VG8010.

O capítulo 2 explica o significado e o modo de usar as diversas instruções MSX-BASIC. Encontrareis uma ainda mais cuidada descrição no manual de referência.

O capítulo 3 (Hardware) explica-vos como alargar todas as possibilidades do vosso computador.

No capítulo 4 (Apêndices) encontrareis informação adicional importante.

Se lerem cuidadosamente este manual, experimentando todos os exemplos dados, em breve podereis escrever os vossos próprios programas no VG8010. Desejamos que se divirtam com o vosso incrivelmente versátil computador VG8010.

Capítulo 1

Instalação do VG8010

1.1 Desembalar

1.2 Fazer as ligações

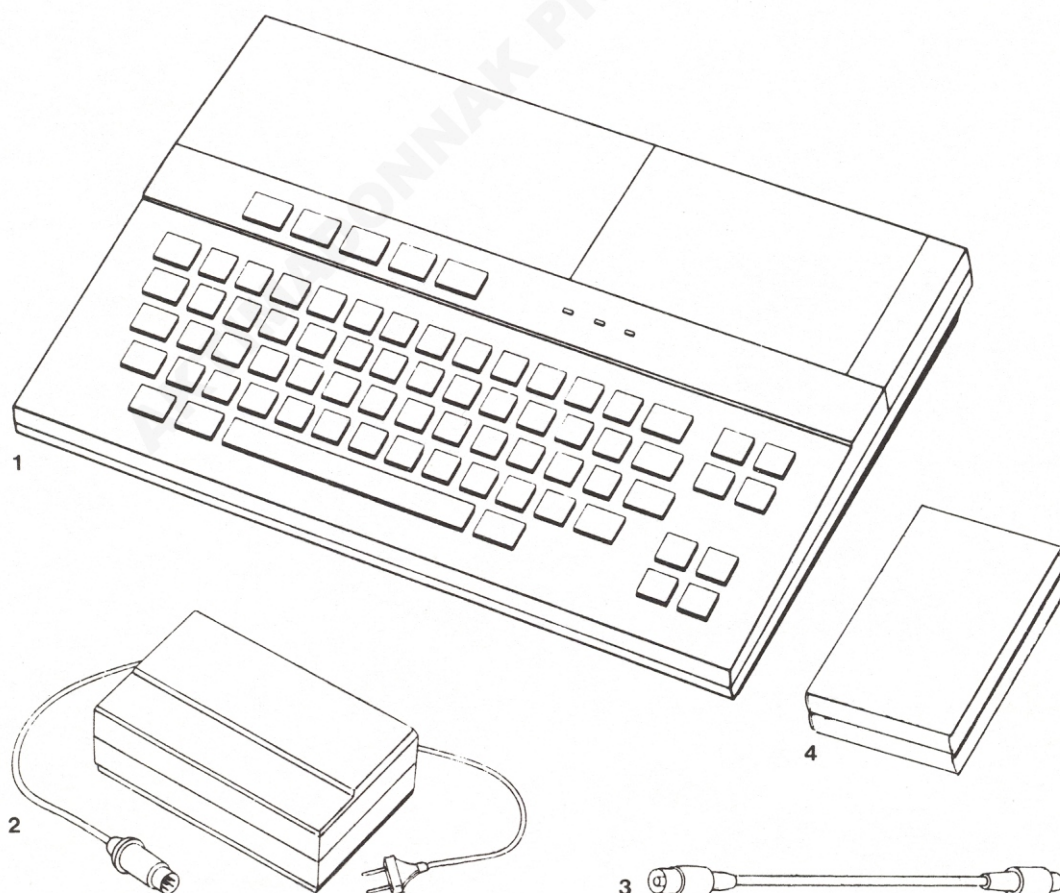
1.3 O teclado

AKTIMADONNAK PRESS 2013

Desembalar

Capítulo 1.1 Ao abrir a caixa, encontrareis tudo o que vos é necessário para começar a trabalhar com o vosso computador MSX. Verifiquem todo o conteúdo. A caixa deve conter o seguinte:

1. A consola VG8010 com um teclado alfanumérico.
2. Uma fonte de alimentação para dar ao vosso computador a voltagem apropriada.
3. Um cabo de ligação ao televisor.
4. Os manuais do VG8010.



Para utilizar o vosso computador VG8010 necessitam apenas de um televisor a preto e branco ou a cores ou monitor. É óbvio que com um televisor a cores ou monitor tirarão todo o proveito das muitas possibilidades da cor.

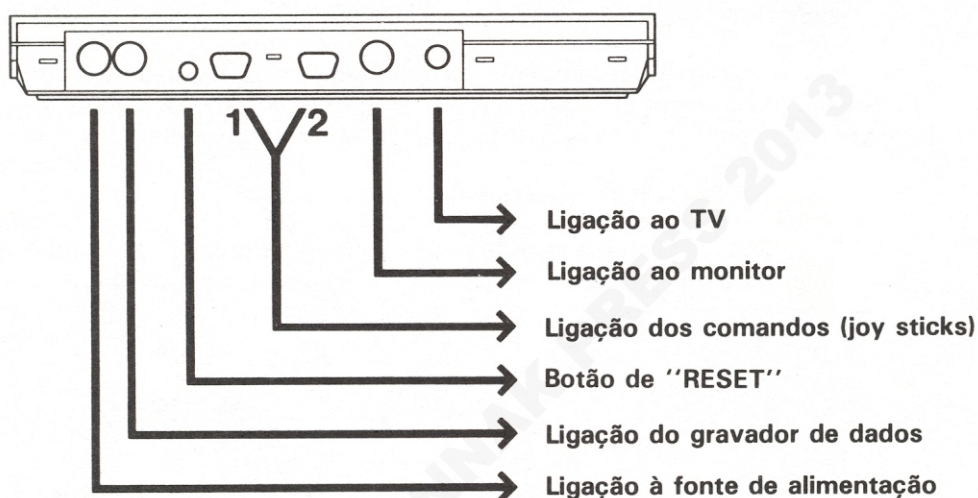
Sigam as instruções de instalação no capítulo 1.2 e eis-vos pronto a começar!

Há uma gama variada de acessórios especiais disponíveis para o vosso computador VG8010 MSX, aumentando desta maneira as suas possibilidades.

1. Um gravador de dados, para ser utilizado como memória externa para armazenar programas, ficheiros, etc.
2. (Joy-sticks) Alavancas de comando para jogar com muitos jogos de video disponíveis para o computador MSX.
3. Um monitor, que vos proporciona uma imagem mais clara, mais nítida... e que evita possíveis conflitos com a família que quer ver o programa normal da TV.
4. Cassette de expansão para aumentar a memória do vosso VG8010.
5. Uma impressora para imprimir os seus programas e constituir-los em arquivo. Em combinação com a impressora e o programa de processamento de palavra, podem também utilizar o conjunto para escrever cartas, relatórios, etc.
6. Variadas cassettes de programas.
7. "Diskettes", com uma grande capacidade e memória externa facilmente acessível, para armazenagem de programas e arquivos.

Como fazer as ligações

Capítulo 1.2 Na parte de trás do vosso computador encontrarão as seguintes fichas de ligação:



Como fazer as ligações

1. Primeiro ligue o cabo especial à ficha TV do seu computador. A outra ponta é então ligada à entrada de antena do televisor.
2. Ligue a unidade de alimentação à tomada do computador. Depois ligue-a à corrente.
3. Ligue agora o computador com o interruptor "on-off" da unidade de alimentação.
4. Ligue o televisor. Escolha um dos canais de pré-selecção do televisor, que não seja usado para recepção normal de programas de televisão e reserve-o para o seu computador. Seleccione o canal correcto (este está indicado num autocolante na parte de trás do computador, próximo da ficha de saída).
5. Verá o seguinte texto no écran:

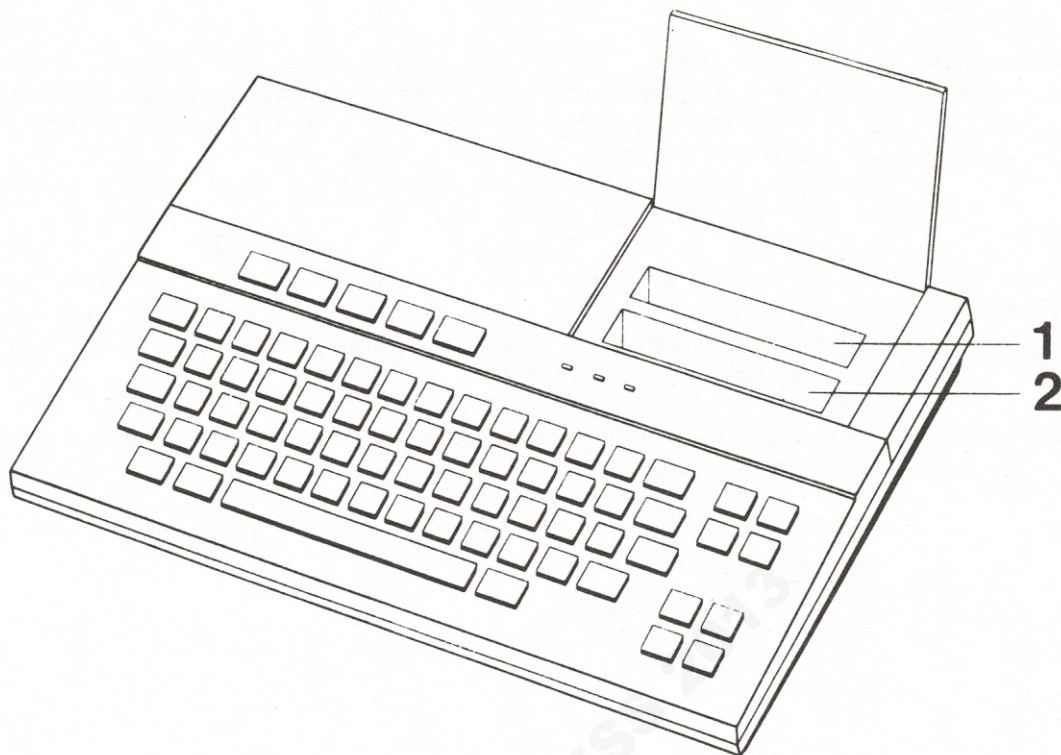
MSX system
etc.

Se estiver a utilizar um televisor a cores ou monitor, o écran será azul e as letras brancas.

6. Quando não há nenhuma cassette inserida vereis apenas o seguinte texto no écran.

MSX BASIC
etc.

O vosso VG8010 está agora pronto a ser usado!



**Como inserir
a cassette**

**DESLIGUE SEMPRE O VG8010 ANTES DE INSERIR UMA CASSETTE NUM DOS
LOCAIS APROPRIADOS!**

Introduza a cassette, com a etiqueta virada para si, numa das aberturas. Ligue agora o computador.

**Como ligar os
comandos
"joy sticks"**

Desligue o computador antes de ligar os comandos. O VG8010 tem tomadas especiais para dois comandos manuais.

A tomada N.º 1 é para o comando direito.

A tomada N.º 2 é para o comando esquerdo.

**Como ligar um
gravador de dados**

Desligue o computador antes de ligar o gravador.

**ISTO SIGNIFICA QUE O GRAVADOR DEVE SER LIGADO ANTES DE COMEÇAR A
ESQUEMATIZAR UM PROGRAMA! SE NÃO PROCEDER ASSIM, TODA A
INFORMAÇÃO ARMAZENADA NA MEMÓRIA DO COMPUTADOR SE PERDERÁ
NO MOMENTO EM QUE A ENERGIA É CORTADA!**

Recomendamos o uso de um gravador de dados com comando remoto.

Quando comprar um gravador de dados assegure-se de que o vendedor lhe fornece também um cabo de ligação apropriado. Ligue este cabo à tomada do gravador no computador. Em muitos casos encontrará três fichas na outra extremidade. Estas devem ser ligadas ao gravador de dados da seguinte maneira:

1. A ficha preta liga-se ao REM do gravador (controlo remoto)
2. A ficha encarnada liga-se ao MIC do gravador.
3. A ficha branca liga-se em EAR do gravador.

Para trabalhar com o gravador de dados ver capítulo 2.5.

**Como ligar
um monitor**

Temos uma tomada especial para ligar o monitor. Um monitor, como certamente sabe, é uma espécie de aparelho de televisão que não recebe os programas regulares da Televisão. No entanto dá-nos imagens duma fonte directa, quer seja dum gravador de vídeo ou dum computador.

Quando comprar um monitor assegure-se de que o vendedor lhe fornece também um cabo de ligação apropriado!

**Como usar
a tecla RESET**

Quando premir a tecla RESET na parte de trás do computador, este ficará temporariamente interrompido. Isto quer dizer que a memória do computador foi completamente apagada e está pronta a começar de novo, tal como quando ligar o computador pela primeira vez.

IMPORTANTE

Verifique se todas as fichas estão convenientemente introduzidas nas tomadas antes de ligar o VG8010.

Para retirar a tampa dos insertores das cassettes abra-a completamente e empurre-a ligeiramente para trás e ela soltar-se-á.

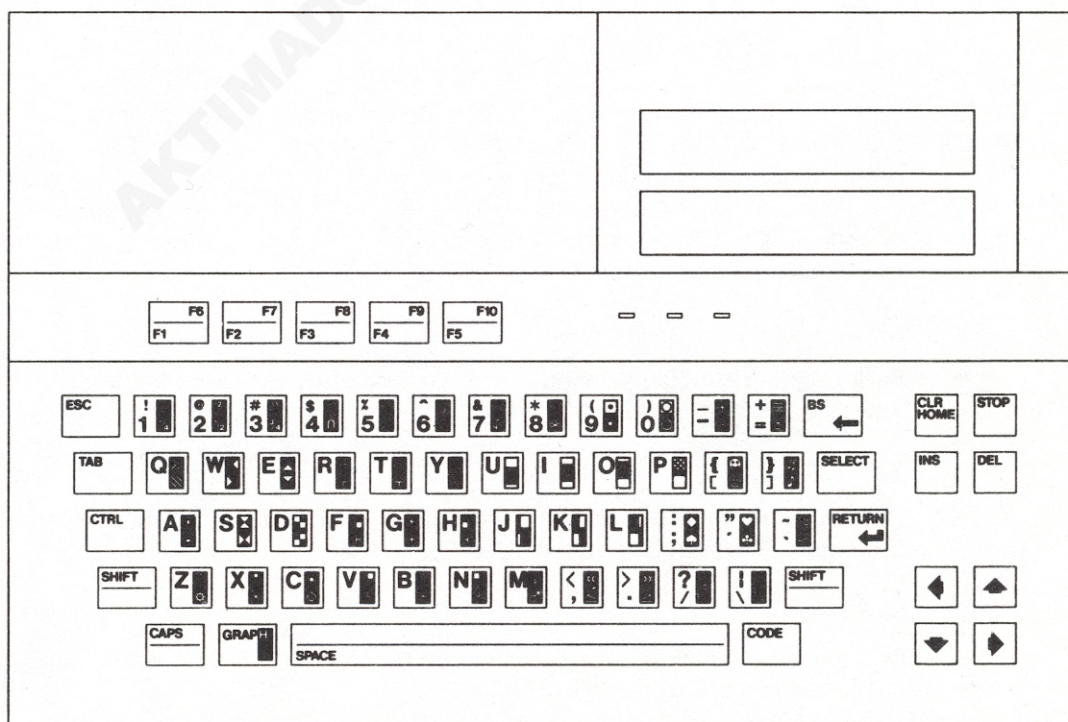
O teclado

Capítulo 1.3 Chegou a altura de experimentar todas as teclas — isto não prejudicará o computador, partindo do princípio de que não seja usada força excessiva. De facto é necessária muita pouca força para premir as teclas. É uma boa ideia verificar quão pouco esforço é preciso para fazer aparecer um carácter no écran.

Se se mantiver uma tecla premida mais tempo do que o necessário, isso produzirá o mesmo efeito que continuar a premir a tecla e o mesmo carácter aparecerá repetidas vezes até que a tecla seja libertada.

O teclado está dividido em quatro grupos de teclas. Começando por cima:

1. Cinco teclas de funções na ficha de cima.
2. No extremo direito encontram-se quatro teclas de comando: "HOME", "STOP", "INS" e "DEL".
3. Imediatamente abaixo das teclas de comando estão as quatro teclas de comando do cursor.
4. A parte restante do teclado inclui todas as teclas da máquina de escrever standard, além de algumas teclas especiais, tais como: "ESC", "TAB", "CTRL", etc.



As teclas de funções

As funções destas teclas são programáveis (ver Capítulo 2.1). No entanto, as teclas estão pré-programadas. As suas funções standard estão indicadas na parte inferior do écran em MSX-BASIC, segundo os indicadores F1 a F5. Se premir a tecla SHIFT verá as funções pré-programadas de F6 a F10 das mesmas teclas. Premindo, por exemplo, simultaneamente as teclas SHIFT e F1, a função F6 é executada.

As teclas de comando

As teclas de comando executam as seguintes funções no computador:

HOME

O cursor desloca-se para a primeira posição da primeira linha quando esta tecla é premida.

HOME + SHIFT

Prima estas duas teclas simultaneamente e o écran ficará limpo de qualquer texto ou imagem.

STOP

Esta tecla é usada para parar um programa. O programa continua se for premida esta tecla de novo. Actua de facto como uma tecla de pausa.

INS

Premindo a tecla INS pode começar-se a função de "inserção" (INSERT). Pode parar-se premindo de novo esta mesma tecla. (INS equivale a INSERT "Inserir").

DEL

Premindo a tecla DEL o character sobre o qual se encontra o cursor é apagado. (DEL equivale a DELETE "Apagar").

Teclas de comando do cursor

As setas indicam a direcção na qual o cursor se deslocará quando qualquer destas teclas é premida.

Teclas de dactilografia

Estas teclas funcionam exactamente como as de uma vulgar máquina de escrever. Se a tecla "A" for premida, a letra "a" aparecerá no écran. Se a mesma tecla for premida simultaneamente com a tecla SHIFT, a letra será um "A" maiúsculo. As outras teclas para algarismos e sinais de pontuação funcionam da mesma maneira. À volta das vulgares teclas para escrever encontram-se as seguintes teclas especiais:

ESC

Esta tecla só funciona quando uma impressora tiver sido ligada ao VG8010 (ESC equivale a ESCAPE "saída").

TAB

Quando esta tecla é premida, o cursor desloca-se para a próxima paragem do tabulador (TAB equivale a TABULATE "tabular").

CTRL

Esta tecla só funciona em combinação com outra, como descrito no Apêndice D (CTRL equivale a "controlo").

SHIFT

A função desta tecla foi descrita no parágrafo precedente.

CAPS

Quando se prime esta tecla o indicador CAPS acende-se. As teclas alfabéticas produzirão apenas caracteres maiúsculos. Se esta tecla for premida de novo as letras serão minúsculas (CAPS equivale a CAPITALS "maiúsculas").

BS

Premindo esta tecla o carácter imediatamente à esquerda do cursor desaparecerá do écran (BS equivale a BACKSPACE "espaço atrás").

SELECT

Esta tecla é apenas utilizada quando se utiliza um programa em cassette. Premindo-se pode fazer-se uma selecção de várias opções que o programa pode oferecer.

RETURN

Esta tecla é utilizada depois de escrever instruções ou ordens em MSX-BASIC.

GRAPH

Quando se prime esta tecla, praticamente todas as teclas produzem um carácter gráfico. O mesmo se aplica quando se prime uma tecla simultaneamente com a tecla SHIFT. Prima de novo a tecla GRAPH, e todas as teclas retomarão as suas funções normais (GRAPH equivale a "graphic").

CODE

Quando se prime esta tecla, o indicador do código acende-se. Todas as teclas produzem agora uma letra específica, "language-bound". O mesmo se aplica quando se prime simultaneamente uma das teclas e a tecla SHIFT. Prima de novo a tecla CODE e todas as teclas retomarão a sua função normal.

Resumindo Praticamente todas as teclas do VG8010 são capazes de produzir 6 caracteres diferentes:

1. Letras minúsculas.
2. Letras maiúsculas (+ SHIFT).
3. Símbolos gráficos 1, depois de premir a tecla de PH.
4. Símbolos gráficos 2 (+ SHIFT) depois de premir a tecla de PH.
5. Caracteres de linguagem "bound" 1 depois de premir a tecla CODE.
6. Caracteres de linguagem "bound" 2 (+ SHIFT) depois de premir a tecla CODE.

Encontrarão um resumo completo no apêndice G deste manual.

Capítulo 2

MSX-BASIC

- 2.1 Como trabalhar com o MSX-BASIC
- 2.2 Como alterar um programa
- 2.3 Constantes e variáveis
- 2.4 Cálculos
- 2.5 MSX-BASIC e o gravador de dados
- 2.6 Tabelas
- 2.7 Decisões
- 2.8 Sub-rotinas
- 2.9 Funções
- 2.10 MSX-BASIC e o écran
- 2.11 Instruções para a cor e para os gráficos
- 2.12 "Sprites" (Movimentação de figuras)
- 2.13 Música
- 2.14 MSX-BASIC e as alavancas de comando (joy sticks)

Como trabalhar com o MSX-BASIC

Capítulo 2.1 Quando se liga o VG8010 MSX, depois de o mesmo ter sido devidamente instalado, verá aparecer o seguinte texto no écran:

```
MSX system  
etc.
```

A seguir o computador verificará se foi colocada alguma cassette no porta-cassettes. Se não foi, o texto seguinte aparecerá automaticamente no écran:

```
MSX BASIC  
etc.
```

O pequeno rectângulo, imediatamente abaixo de "Ok" é chamado o "cursor". Indica onde aparecerá qualquer texto que você escreva.

O número indica o número de posições que ainda estão livres na memória do computador.

O "Ok" aparecerá no écran sempre que o MSX-BASIC está ao nível de comando, o que quer dizer que está pronto para as vossas instruções.

Neste ponto podem escolher dois modos de usar o MSX-BASIC: o "Modo Directo" e o "Modo Indirecto".

O modo directo Trabalhando com o modo directo, todos os comandos e praticamente todas as instruções (com ou sem funções) podem ser utilizadas. Encontrará um sumário completo de todos os comandos, instruções e funções no manual de referências. No modo directo, todos os comandos e instruções são imediatamente executados logo que a tecla RETURN seja premida. Não são armazenados na memória do computador.
Escreva o exemplo seguinte:

```
PRINT 5+2
```

Se cometer um erro ao escrever, use a tecla BS (Back Space) para apagar o caracter imediatamente à esquerda do cursor. Deste modo podem apagar-se tantas letras quantas forem necessárias para fazer a devida correcção. Quando a linha está correcta, prime-se a tecla RETURN.

É importante lembrar-se a partir de agora de premir esta tecla depois de ter acabado de escrever um comando ou uma instrução.

MSX-BASIC executará imediatamente a instrução e a resposta aparecerá no écran:

```
7  
Ok
```

O "OK" quer dizer que o "MSX-BASIC" voltou ao nível de comando, aguardando a próxima instrução.

O modo directo Trabalhando com o modo indirecto, todas as instruções são precedidas por um número de linha. Não serão executadas imediatamente após premida a tecla RETURN mas são armazenadas na memória do computador em sequência, de acordo com o número da linha.
Escreva agora este exemplo:

```
10 PRINT 5+2
```

Prima a tecla RETURN e verá que nada acontece, excepto que o cursor salta para a primeira posição da linha seguinte, aguardando novas instruções. Isto significa também que a linha 10 está agora armazenada na memória do computador. Uma combinação de tais linhas, contendo instruções, etc. chama-se um programa. Todas as instruções do programa serão eventualmente executadas em sequência, de acordo com o número da linha quando é dada uma ordem "RUN" e é premida a tecla RETURN.

Trabalhando no modo indirecto, cada nova linha tem que começar com um número de linha. Tem muito por onde escolher porque pode utilizar qualquer número entre 0 e 65529. É aconselhável não usar números sucessivos quando se escreve um programa, mas deixar alguns números entre as linhas. Procedendo assim podem acrescentar-se em qualquer altura linhas adicionais.

Também pode pedir-se ao MSX que se encarregue da numeração. Inicie com a instrução "AUTO" antes de começar a programação. Prime-se RETURN e o MSX mostra automaticamente 10 como a primeira linha do programa. O asterisco depois de 10 indica que a linha número 10 já está na memória do computador. Premindo a tecla RETURN o número seguinte será automaticamente 20, etc. Para parar o processo de numeração automática devem premir-se simultaneamente as teclas CTRL e C.

MSX-BASIC permite o uso de um máximo de 255 caracteres (incluindo espaços e número de linha) por linha de programa. Uma linha de programa pode conter várias instruções, desde que estejam separadas por dois pontos (:). Tentem este exemplo:

```
10 PRINT 8+2:PRINT 6*4  
RUN  
10  
24  
Ok
```

MSX-BASIC fez os dois cálculos separadamente, visto estarem separados por dois pontos.

Para tirar todas as linhas prévias do programa da memória do computador, use o comando "NEW" antes de premir de novo a tecla RETURN.

Diferenças entre comandos e instruções Ordens e instruções podem ser usadas tanto no modo directo como no indirecto. Quando uma ordem é executada, o MSX-BASIC voltará sempre ao nível de comando. Tente escrever o seguinte exemplo:

```
10 PRINT 3+1
20 LIST
30 PRINT 6-3
RUN
```

"LIST" é um comando MSX-BASIC e é usado para projectar todas as linhas de um programa no écran. É uma ordem usada para rever ou verificar um programa e para fazer modificações ou juntar mais qualquer coisa a um programa.

Porque "LIST" é uma ordem, MSX-BASIC voltará ao nível de comando depois de ter executado a linha 20 deste exemplo. Isto significa que a linha 30 não será executada.

No manual de referência MSX-BASIC pode certificar-se se uma declaração é uma ordem ou uma instrução.

A diferença entre instruções e funções Uma função nunca pode ser usada sem ser precedida duma instrução. "FRE(0)" é exemplo de uma função. É utilizada para perguntar ao MSX-BASIC quantos bytes estão ainda disponíveis na memória do computador. Experimentemos o exemplo seguinte. A ordem "NEW" é utilizada para remover todas as linhas previamente escritas na memória do computador:

```
NEW
10 FRE(0)
```

Agora dê a ordem RUN e verificará que o MSX-BASIC apresenta uma mensagem de erro no écran: "Syntax error in 10". Isto acontece porque foi usada uma função sem primeiro juntar a instrução necessária. Experimente o exemplo:

```
10 PRINT FRE(0)
RUN
```

Agora o MSX-BASIC indicará quantos bytes estão ainda disponíveis para continuar o programa.

Como escrever as instruções "PRINT" é uma das instruções que se encontram com mais frequência. Para poupar tempo pode usar-se o ponto de interrogação. MSX-BASIC transformará o ponto de interrogação na instrução "PRINT". Vamos ver como isto funciona. Escreva:

```
10 ? 2+9
```

Prima a tecla RETURN e escreva a ordem "LIST". Agora prima a tecla RETURN novamente e verá:

```
10 ? 2+9
LIST
10 PRINT 2+9
```


MSX-BASIC permite escrever ordens e instruções utilizando apenas uma tecla. As teclas de funções de F1 a F10 foram reservadas para este fim. Na parte inferior do écran pode ver-se quais as funções das teclas F1 e F5. Premindo a tecla SHIFT pode ler-se no écran quais as ordens e instruções que são activadas com as teclas F6 e F10.

Pode também pedir-se ao MSX-BASIC para dar uma listagem das ordens e instruções que foram programadas nas teclas de funções. Escreve-se a instrução "KEY LIST" e prime-se na tecla RETURN.

Podem também alterar-se as ordens e instruções utilizando a instrução "KEY". Tentemos o seguinte exemplo:

```
KEY 1,"GOSUB"
```

A seguir premimos a tecla RETURN. De agora em diante a tecla F1 produzirá a instrução "GOSUB".

Pode retirar-se do écran a referência programada para as teclas de funções com a instrução "KEY OFF" e dá-se precisamente o inverso se utilizarmos a instrução "KEY ON".

**Quando e como
utilizar """:;,.**

MSX-BASIC executará todos os pedidos desde que devidamente escritos. Isto também se aplica às pontuações. Todas têm uma função específica quando se escreve um programa.

Todos os caracteres, por exemplo entre aspas, aparecerão no écran tal qual foram escritos. Pode verificar-se isto escrevendo os dois exemplos seguintes:

```
10 PRINT 5+2
20 PRINT "5+2"
RUN
```

No primeiro caso o MSX-BASIC faz o cálculo $5 + 2 = 7$, mas no segundo reproduzirá o texto tal qual como foi escrito, isto é, $5 + 2$.

Os dois pontos (:) servem para dividir duas instruções numa linha de programa, como já mencionámos atrás.

O ponto e vírgula (;) pode ser usado numa instrução "PRINT". Depois de executada a instrução, ao chegar ao ponto e vírgula o cursor não segue imediatamente para a linha seguinte.

O ponto e vírgula é também usado em combinações com a instrução "INPUT" quando a situação requer uma frase interrogativa.

O ponto final (.) pode ser usado em conjunto com as ordens "LIST" "AUTO" e "DELETE" se nos quisermos referir à última linha do programa.

A vírgula (,) pode apenas ser usada em combinação com ordens e instruções como se descreve no manual de referência MSX-BASIC.

**Mensagem
com erro**

Quando o MSX-BASIC encontra um erro numa linha dum programa, exibirá no écran uma mensagem com erro. Encontra-se no Apêndice A deste manual um sumário completo das mensagens com erro.

Como alterar um programa

Capítulo 2.2 Uma linha de um programa pode ser facilmente alterada escrevendo uma nova linha com o mesmo número de linha.
Novas linhas podem ser acrescentadas a um programa introduzindo novas linhas com números de linha que ainda não foram usados, escolhendo sempre um número de linha que coloque a nova linha na sequência própria.
Podem ser apagadas linhas de programa com a ordem "DELETE". Se quiser apagar somente uma linha basta escrever o número da linha, premindo depois a tecla RETURN.
Também é possível mudar no écran linhas de programa. A isto chama-se "screen editing" (inserções no écran)

"Screen editing"
(inserções no écran) "Screen editing" significa que as linhas de programa que aparecem no écran podem ser alteradas com as teclas INS, BS e DEL.
Tentemos o exemplo seguinte:

```
NEW  
10 pontt
```

e prima-se a tecla RETURN. O que de facto pretendíamos escrever era "PRINT" e fizemos um erro de escrita. Não nos preocupemos por ter usado letras minúsculas. MSX-BASIC converterá automaticamente todas as ordens, instruções e funções em maiúsculas. Agora escrevemos:

```
LIST
```

e premimos a tecla RETURN. No écran aparecerá:

```
10 PONTT
```

Uma vez que queremos que PONTT seja alterado para "PRINT", temos que alterar esta linha de programa. Primeiro dirigimos o cursor para a letra "O" de "PONTT" utilizando as teclas do cursor. A seguir premimos a tecla INS (insert). Verificamos que o cursor se transformou numa faixa estreita. Isto significa que cada letra ou número que se escreva agora será inserido entre as letras já no écran. Prima-se a letra "R" e veremos que esta letra foi colocada entre o "P" e o "O". Premimos a tecla INS outra vez. O cursor voltará agora ao seu tamanho normal. Isto quer dizer que qualquer letra ou número que se escreva agora substituirá o símbolo que estiver no écran. Prima-se a tecla "I" e a linha agora será:

```
10 PRINTT
```


O cursor cobre o "N". O que temos que fazer agora é apagar um "T". Deslocamos o cursor para o primeiro "T" com as teclas de controle do cursor. Agora prime-se a tecla DEL (delete — apagar). O primeiro "T" foi apagado e a outra letra "T" deslocou-se um espaço para a esquerda, preenchendo a falha. Prima-se de novo a tecla RETURN.

As alterações só entram no MSX-BASIC quando a tecla RETURN é premida.

Quando se escreve a ordem "LIST" seguida de RETURN, MSX-BASIC mostrar-nos-á que a linha ficou:

```
10 PRINT
```

que é exactamente o que desejávamos.

Temos de nos certificar sempre de que novos comandos ou instruções depois de "screen editing" entram numa linha vazia no écran. Suponhamos que temos, por exemplo, o seguinte texto no écran:

```
10 PRINT 9+2
20 PRINT 3-6
```

Vamos supor agora que o cursor cobre o "2" da linha de programa 20, já que acabámos de alterar a linha 10. Se agora escrevermos a ordem "LIST" obteremos este resultado:

```
10 PRINT 9+2
LISTRINT 3-6
```

Ao premirmos a tecla RETURN, o MSX-BASIC indicará um erro de mensagem, uma vez que a ordem "LISTRINT 3-6" não existe em MSX-BASIC. Pode evitar-se isto, claro, deslocando o cursor para uma linha vazia no écran ou premindo as teclas "SHIFT" e "HOME" simultaneamente antes de dar o comando "LIST".

Se o conteúdo dessa linha não tiver grande importância, pode também premir-se simultaneamente as teclas CTRL e E. Isto apagará todos os símbolos dessa linha, seguindo a ordem "LIST".

A tecla STOP A tecla STOP não tem qualquer função quando o MSX-BASIC está no nível de comando.

Quando o MSX-BASIC está a executar um programa, podemos pará-lo premindo a tecla STOP. Ao mesmo tempo o cursor aparecerá de novo no écran. O programa continuará quando se premir a tecla STOP novamente. Se se quiser interromper um programa e voltar ao nível de comando do MSX-BASIC, primam-se simultaneamente as teclas CTRL e STOP. No écran aparecerá a seguinte mensagem:

```
Break in nnnnn
```

Aqui os "nnnnn" equivalem ao número da linha em que o programa foi interrompido.

Constantes e variáveis

Capítulo 2.3 A memória do MSX é capaz de armazenar mais do que apenas linhas com ordens, instruções e funções. Há outros elementos que têm um papel importante durante a execução dum programa. Podem ser divididos em "constantes" e "variáveis".

Constantes As constantes contêm informações que não se alteram durante o programa. Há duas espécies de constantes: constantes alfanuméricas e numéricas. Uma constante alfanumérica pode conter com um máximo de 255 caracteres e tem que começar e acabar com aspas ("..."). Exemplo: "HELLO" e "12345". Apesar da constante "12345" conter números, não pode ser utilizada para fazer cálculos. Uma constante numérica é de facto um número positivo ou negativo. A vírgula decimal tem que ser substituída por um ponto (.) em MSX-BASIC. Exemplo: 123,45 escreve-se: 123.45 em MSX-BASIC.

O MSX-BASIC reconhece seis diferentes espécies de constantes numéricas:

1. Constante de número inteiro

Isto é qualquer número entre -32768 e 32767
Exemplo: 13 ou -25

2. Constante com parte decimal

Número com algarismos depois do ponto decimal.
Exemplo: 15.73 ou -178.3

3. Constante com notação científica

Número escrito com notação científica.
Exemplo: 2.34E8 ($= 2.34 * 10^8 = 234,000,000$)

4. Constante hexadecimal

É o correspondente hexadecimal ao número dado. É precedido por "&H"
Exemplo: &H3F

5. Constante octal

É um número de base oito. É precedido por "&O".
Exemplo: &O37

6. Constante binária

É um número binário e é precedido por "&B".
Exemplo: &B01010111

Variáveis Uma variável é uma parte da memória que serve para armazenar um item de informação para o computador. A cada variável é dado um nome pelo programador, sendo o computador capaz de a encontrar quando necessário. Há duas espécies diferentes de variáveis: alfanuméricas e numéricas.

O nome de uma variável alfanumérica termina sempre com o sinal de dólar (\$), "STRING". Exemplo: A\$ = "HELLO" ou AB\$ = "12345" lê-se "Astring" ou "ABstring".

Uma variável numérica pode conter quer um número positivo quer um negativo. Uma variável numérica inteira é uma variável que só pode conter um número entre -32768 e 32767. O nome de tal variável termina sempre com o sinal de (%). Exemplo: A% = 10

Precisão simples e dupla As constantes numéricas e as variáveis podem ser dadas em precisão simples e dupla. Numa **forma de precisão simples** os números são armazenados em 4 bytes da memória. Tal número pode contar um máximo de 6 algarismos.

Uma constante de precisão simples é indicada por um ponto de exclamação (!). Exemplo: 12.5!

Uma constante de precisão simples, quando de vírgula flutuante, indica-se com um "E" como símbolo exponencial. Exemplo: 2.34E8

Uma variável com precisão simples é indicada por um ponto de exclamação (!) como o último carácter do nome. Exemplo: AB!

Na **forma de precisão dupla** os números são armazenados num espaço de 8 bytes da memória. Um tal número pode conter um máximo de 14 algarismos.

Uma constante de precisão dupla é indicada pelo símbolo "#". Exemplo: 23.33#

Uma constante de precisão dupla e vírgula flutuante é indicada por um "D" como símbolo da base. Exemplo: 3.768D-7

Uma variável de precisão dupla é indicada pelo símbolo "#" como o último carácter do nome. Exemplo: DA#

Se o nome para uma variável não tiver "\$", "%", "!" ou "#" como último carácter, o MSX-BASIC tomará essa variável como uma variável de precisão dupla.

Exemplo:

ME# é uma variável de precisão dupla, tomando 8 bytes da memória.

PT% é uma variável com números inteiros, tomando 2 bytes na memória.

ZS! é uma variável de precisão simples, tomando 4 bytes na memória.

ME é uma variável de precisão dupla, tomando 8 bytes na memória.

AB\$ é uma variável alfanumérica, tomando até 3 bytes na memória mais o espaço requerido pelo byte para o conteúdo da variável.

MSX-BASIC permite que um total de 200 caracteres sejam armazenados em forma de variáveis alfanuméricas na memória do computador. A instrução "CLEAR" torna possível alterar este máximo de 200. Com o MSX-BASIC o tipo de variável

pode ser também determinado com as instruções "DEFINT", "DEFSTR", "DEFSNG" e "DEFDBL". Estas instruções estão pormenorizadas no Manual de Referência.

Conversões MSX-BASIC converterá, se necessário, variáveis numéricas em diferentes tipos de constantes.

Exemplo 1:

```
NEW
10 A%=13.03
20 PRINT A%
RUN
13
```

Neste exemplo uma constante com parte decimal é inserida numa variável inteira. Os algarismos a seguir ao decimal são apagados.

Exemplo 2:

```
NEW
10 E#=6!/7!
20 PRINT E#
RUN
.85714285714286
```

Embora o cálculo "6!/7!" esteja indicado como de precisão simples, o resultado é representado com dupla precisão uma vez que a variável "E#" é dada com dupla precisão.

Exemplo 3:

```
NEW
10 E!=6/7
20 PRINT E!
RUN
.857143
```

Embora o cálculo "6/7" seja dado com dupla precisão, o resultado vem com precisão simples, uma vez que a variável "E!" é dada com precisão simples.

Exemplo 4

```
NEW
10 B!=SQR(2)
20 A=B!
30 PRINT B!;A
RUN
1.41421 1.41421
```

Apesar do facto da variável "A" ser de precisão dupla, apenas 6 algarismos são mostrados, uma vez que o conteúdo da variável (A) de precisão dupla foi tornado idêntico ao da variável (B!) de precisão simples.

A conversão de vários tipos de variáveis pode também ser atingida com as funções "CDBL", "CINT", "CSNG" e "STR\$". No Manual de Referência encontram-se mais pormenores sobre estas funções.

Nomes de variáveis O programador pode escolher ele próprio os nomes das variáveis, desde que recorde as seguintes condições:

1. O nome deve sempre começar com um carácter alfanumérico (A a Z) enquanto que os outros caracteres podem ser quer numéricos quer alfabéticos.
2. Embora um nome possa consistir de mais de dois caracteres, o MSX-BASIC apenas reconhece os dois primeiros caracteres. Os nomes "PO" e "POST", por exemplo, referem-se a uma mesma variável.
3. O nome da variável não pode ser nenhuma das palavras reservadas apresentadas no Apêndice F deste manual.

Conteúdo das variáveis Pode ser dado um certo valor a uma variável usando a instrução "LET".
Exemplo:

```
10 LET A=10
20 LET B$="HELLO"
```

O uso da palavra "LET" não é estritamente necessário. Obter-se-á o mesmo efeito sem a utilizar.

Pode dar-se um conteúdo a uma variável utilizando a instrução "INPUT". Quando o computador está a executar uma instrução "INPUT" espera até que todas as informações tenham sido escritas no teclado. Depois de premida a tecla "RETURN" a informação é colocada na variável indicada pela instrução "INPUT". Uma frase interrogativa pode ser acrescentada utilizando a instrução "INPUT".
Exemplo:

```
NEW
10 INPUT "GIVE A NUMBER";A
20 INPUT "YOUR NAME";B$
RUN
GIVE A NUMBER?
```

O ponto de interrogação é acrescentado automaticamente. A linha 10 do programa MSX-BASIC aceita apenas informação numérica. Na linha 20 podem ser utilizados símbolos alfanuméricos, entre aspas.

Variáveis especiais Com o MSX-BASIC podem utilizar-se variáveis especiais ao escrever um programa. Estas variáveis são "TIME", "VDP", "BASE" e "SPRITE\$".
Exemplo:

```
NEW
10 PRINT "START":TIME=0
20 FOR I%=1 TO 1000:NEXT I%
30 PRINT "IT IS NOW";TIME/50;"SECONDS LATER"
RUN
START
IT IS NOW .8 SECONDS LATER
```

O exemplo acima mostra que o computador leva .8 (0,8) de segundo para executar a instrução "FOR ... NEXT" na linha 20 do programa.

A variável especial "SPRITE\$" é explicada detalhadamente no capítulo 2.12. As variáveis "VDP" e "BASE" devem apenas ser usadas por aqueles que estão familiarizados com o modo de funcionamento do circuito de Vídeo (video chip).

Cálculos

Capítulo 2.4 O MSX-BASIC reconhece cinco operações aritméticas:

1. Soma (+)
2. Subtração (—)
3. Multiplicação (*)
4. Divisão (/)
5. Exponenciação (^)

Estas operações podem ser combinadas com uma instrução, por exemplo assim:

```
NEW
10 PRINT 40/5+3
RUN
11
```

Devemos lembrar-nos que uma operação aritmética é executada seguindo esta ordem:

1. Exponenciação
2. Multiplicação e divisão
3. Soma e/ou subtração

Esta ordem pode, evidentemente, ser alterada, utilizando parentesis como no exemplo seguinte:

```
NEW
10 PRINT 40/(5+3)
RUN
5
```

Divisão de números inteiros

MSX-BASIC pode dividir pelo método dos inteiros, o que significa que a divisão será executada com os números completos e que o resultado será também dado em números completos. Uma divisão de inteiros é indicada por um "back slash". Tentemos primeiro a divisão normal:

```
NEW
10 PRINT 10/4
RUN
2.5
```


Repare-se na diferença quando tentamos o exemplo seguinte:

```
NEW
10 PRINT 10\4
RUN
2
```

O resultado é sempre dado em número inteiro, desprezando o resto.

Cálculo de Módulos

Um cálculo em módulos dá o valor residual numa expressão de números inteiros depois duma divisão com inteiros. Um módulo de cálculo é instituído pela palavra "MOD". Por exemplo:

```
NEW
10 PRINT 25.68 MOD 6.99
RUN
1
```

Neste exemplo os números 25.68 e 6.99 são transferidos em inteiros (25 e 6) após o que a divisão é executada. O quociente é 4 com um valor residual de 1.

Divisão por zero e "overflow" (exceder a capacidade de cálculo)

Se se faz uma divisão por zero durante a execução de um cálculo, o MSX-BASIC dará uma mensagem de erro e pára o programa.

Se o campo de recepção for demasiado pequeno para conter o resultado dum cálculo, o MSX-BASIC dará também uma mensagem de erro e parará a execução do programa.

Variáveis

Se a variável é usada numa operação matemática, a operação é executada com o conteúdo dessa variável. Por exemplo:

```
NEW
10 A=10:B=5
20 C=A*B
30 PRINT C
RUN
50
```

Na linha 20 do programa, dez (o conteúdo de "A") é multiplicado por cinco (conteúdo de "B").

Não podem ser feitas operações matemáticas com dados alfanuméricos. É no entanto possível acasalar variáveis alfanuméricas.

Exemplo:

```
NEW
10 B$="MSX ":C$="- COMPUTER"
20 A$=B$+C$
30 PRINT A$
40 PRINT B$+"- BASIC"
RUN
MSX - COMPUTER
MSX - BASIC
```

Neste caso o conteúdo da "A\$" é "MSX-COMPUTER".

Funções matemáticas

Um certo número de cálculos matemáticos frequentemente usados são representados em MSX-BASIC como uma função:

A função "ABS" refere-se ao valor absoluto dum algarismo.

A função "COS" dá o co-seno dum ângulo em termos de radianos.

A função "EXP" dá o exponencial dum algarismo.

A função "LOG" dá o logaritmo de um algarismo.

A função "SIN" dá o seno dum ângulo em termos de radianos.

A função "SQR" dá a raiz quadrada de um algarismo.

A função "TAN" dá a tangente dum ângulo em termos de radianos.

Tentemos este exemplo:

```
NEW
10 X=25
20 PRINT X;SQR(X)
RUN
25 5
```

Outras funções matemáticas podem ser compostas como é indicado no Apêndice B. O programador pode também definir outras funções matemáticas com a instrução "DEF FN". Tentaremos o exemplo seguinte:

```
NEW
10 DEF FNAB(X,Y)=X^3/Y^2
20 X=6:Y=2
30 T=FNAB(X,Y)
40 PRINT T
RUN
54
```

Os valores de X e Y, como é indicado na linha 20 do programa, são substituídos pela defenição matemática da linha 10 na linha 30. A variável "T" contém o resultado de $6^3/2^2$.

MSX-BASIC e o gravador de dados

Capítulo 2.5 O MSX-BASIC permite o uso de um gravador de dados para armazenar programas e arquivá-los numa cassette compacta. Mais tarde podem ser lidos através da memória do computador.

Como gravar programas na cassette Um programa pode ser transferido para uma cassette através da ordem "CSAVE".
Procede-se do seguinte modo:

1. Coloca-se uma cassette virgem no gravador e rebobina-se completamente.
2. Bobina-se a cassette para a frente até ter passado a faixa guiadora.
3. Prime-se simultaneamente as teclas PLAY e REC do gravador.
4. Escreve-se a ordem "CSAVE" imediatamente seguida do nome do programa que se vai introduzir na cassette. Coloque o nome do programa entre aspas (CSAVE "nome do programa")
5. Prime-se a tecla RETURN.

O programa será transferido para a cassette.

Como ler um programa a partir duma cassette Um programa pode ser lido da cassette para a memória do computador através da ordem "CLOAD".

1. Inserir a cassette que contém o programa no gravador e rebobiná-la completamente.
2. Premir a tecla PLAY do gravador.
3. Escreva a ordem: CLOAD "nome do programa".
4. Premir a tecla RETURN.

Se não for indicado o nome do programa juntamente com a ordem "CLOAD" o primeiro programa a aparecer será lido pela memória do computador. Quando o MSX-BASIC encontra um nome diferente de programa daquele que se pediu, o texto seguinte aparecerá no écran:

SKIP (PROGRAM NAME)

O MSX-BASIC continuará a procurar o programa certo. Se o programa foi encontrado, o texto seguinte aparecerá no écran e o programa será lido para a memória do computador.

FOUND (PROGRAM NAME)

Quando a tarefa estiver completada, MSX-BASIC voltará ao nível de comando.

**Como gravar
dados
numa cassette**

As informações colhidas enquanto se utiliza um programa, podem também ser armazenadas em cassette. As informações são gravadas em sequência. São referenciadas como um ficheiro. Cada ficheiro tem um nome e tem de ser aberto. O número de ficheiro dentro dum programa tem de ser especificado com a instrução "MAXFILES". Aqui está um exemplo:

```
5 MAXFILES=1
10 OPEN "CAS:ADDR" FOR OUTPUT AS#1
20 A=15:B=23.5:C=10:D=25.2:E=13
30 PRINT #1,A;B;C
40 PRINT #1,D;E
50 CLOSE #1
```

Neste exemplo, na linha 10, o nome do ficheiro "ADDR" é aberto (aqui referimo-nos a um ficheiro de endereços, mas qualquer nome serve, dependendo, claro, da espécie de ficheiro).

A palavra "CAS" indica que o arquivo está gravado numa cassette.

A informação é de facto gravada numa cassette com a instrução "PRINT #".

Depois de cada instrução "PRINT #", um código "CR" e "LF" é colocado na fita.

Uma vírgula é automaticamente colocada entre as variáveis numéricas. O nosso exemplo acima citado dará a seguinte "imagem" na fita:

1	5	,	2	3	.	5	,	1	0	C	R	L	F	2	5	.	2	,	1	3	C	R	L	F
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Ao escrever uma variável alfanumérica, a vírgula (,) deve ser acrescentada manualmente com a instrução "PRINT #", como no seguinte exemplo:

```
5 MAXFILES=1
10 OPEN "CAS:ADDR" FOR OUTPUT AS#1
20 A=15:B=23.5:C=10:D=25.2:E=13
30 PRINT #1,A;B;C
40 PRINT #1,D;E
50 CLOSE #1
```

Isto produzirá o seguinte resultado na fita:

J	O	H	N	,	P	E	T	E	C	R	L	F
---	---	---	---	---	---	---	---	---	---	---	---	---

Devemos assegurar-nos de que os pontos de procedimento de 1 a 3, como explicado para a gravação de um programa, são executados antes de transferirmos os dados para a cassette!

**Como ler
informações
duma cassette**

Dados gravados numa cassette podem ser lidos pela memória do computador através da instrução "INPUT #".

Para ler as variáveis numéricas do nosso exemplo anterior as seguintes instruções devem ser usadas:

```
5 MAXFILES=1
10 OPEN "CAS:ADDR" FOR INPUT AS #1
20 INPUT #1;V,W,X
30 INPUT #1;X,Y
40 CLOSE #1
50 CLOSE #1
```


Para variáveis alfanuméricas são precisas as seguintes instruções, por exemplo:

```
5 MAXFILES=1
10 OPEN "CAS:ADDR" FOR INPUT AS #1
20 INPUT #1;V,W,X
30 INPUT #1;X,Y
40 CLOSE #1
50 CLOSE #1
```

Uma instrução "INPUT#", preenche a variável com a informação até à primeira vírgula ou os códigos "CR" e "LF".

Uma instrução "LINE INPUT#" preenche a mesma variável com a informação até aos próximos códigos "CR" e "LF", como no exemplo:

```
5 MAXFILES=1
10 OPEN "CAS:ADDR" FOR INPUT AS #1
20 INPUT #1;V,W,X
30 INPUT #1;X,Y
40 CLOSE #1
50 CLOSE #1
```

Neste exemplo "X\$" contém "JOHN, PETE".

Com a função "EOF" pode perguntar-se, se chegou ao fim do ficheiro.

Devemos assegurar-nos de que seguimos cuidadosamente os pontos de procedimento de 1 a 2, como especificado nas instruções para ler uma cassette, antes de ler a informação para a memória do computador!

Arquivos de endereços

Este programa pode ser usado para arquivar nomes, endereços, etc. A informação é transferida para a cassette e pode ser lida em qualquer altura.

```
NEW
10 SCREEN 0:KEY OFF:WIDTH 39:MAXFILES=1
20 CLS:LOCATE 10,2:PRINT "MENU"
30 ON KEY GOSUB 100,400,700
40 LOCATE 5,5:PRINT "F1=ADDRESS INPUT"
50 LOCATE 5,7:PRINT "F2=ADDRESS PRINT"
60 LOCATE 5,9:PRINT "F3=END PROGRAM"
70 KEY(1) ON:KEY(2) ON:KEY(3) ON
80 GOTO 80
100 CLS
110 PRINT:PRINT "PUT YOUR DATA RECORDER AT READY AND
PRESS THE PLAY AND RECORD KEYS SIMULTANEOUSLY"
120 PRINT:PRINT:INPUT "READY (Y/N)";F$
130 IF F$="N" GOTO 120
140 IF F$="n" GOTO 120
150 OPEN "CAS:ADDRESS" FOR OUTPUT AS#1
160 CLS
170 LOCATE 1,5:INPUT "NAME";A$
180 LOCATE 1,7:INPUT "ADDRESS";B$
190 LOCATE 1,9:INPUT "TOWN";C$
200 LOCATE 1,11:INPUT "TELEPHONE";D$
210 LOCATE 1,20:INPUT "CORRECT (Y/N)";F$
```



```

220 IF F$="N" GOTO 160
230 IF F$="n" GOTO 160
240 PRINT #1,A$;" ";B$;" ";C$;" ";D$
250 LOCATE 1,22:INPUT "ANOTHER ADDRESS (Y/N)";F$
260 IF F$="N" GOTO 290
270 IF F$="n" GOTO 290
280 GOTO 160
290 CLOSE #1:RETURN 20
400 CLS
410 PRINT:PRINT "PUT YOUR DATA RECORDER AT READY AND
PRESS THE PLAY KEY"
420 PRINT:PRINT:INPUT "READY (Y/N)";F$
430 IF F$="N" GOTO 420
440 IF F$="n" GOTO 420
450 OPEN "CAS:ADDRESS" FOR INPUT AS#1
460 CLS:PRINT
470 IF EOF(1) THEN 570
480 INPUT #1,A$,B$,C$,D$
490 PRINT "NAME      ":";A$:PRINT
500 PRINT "ADDRESS   ":";B$:PRINT
510 PRINT "TOWN      ":";C$:PRINT
520 PRINT "TELEPHONE ":";D$
530 LOCATE 1,22:INPUT "NEXT ADDRESS (Y/N)";F$
540 IF F$="N" GOTO 570
550 IF F$="n" GOTO 570
560 GOTO 460
570 CLOSE #1:RETURN 20
700 CLS:PRINT "END PROGRAM"
710 END

```

Corra o programa e tente introduzir alguns endereços. Serão gravados na fita ao premir a tecla F1 e são transmitidos para a memória do computador ao premir a tecla F2. Não esquecer de rebobinar a cassette antes de tentar ler os endereços da cassette.

Se o programa não funcionar devidamente, deve verificá-lo cuidadosamente pois podem existir erros de escrita!

Outras instruções para lidar com as cassettes são: BLOAD, BSAVE, LOAD "CAS:", SAVE "CAS:", MERGE, MOTOR. Ver o Manual de Referência.

Para carregar software pré-gravado, ler as instruções de operação para esse software.

Tabelas

Capítulo 2.6 Quando muitos dados são constantemente usados num programa, é aconselhável registá-los numa tabela usando a instrução "DATA". Os dados são obtidos da tabela através da instrução "READ", como no exemplo seguinte:

```
NEW
10 READ X$:PRINT X$
20 GOTO 10
30 DATA JAN,FEB,MAR
RUN
JAN
FEB
MAR
Out of DATA in 10
```

Note-se que os items de informação com a instrução "DATA" são separados por uma vírgula (,). A instrução "READ" vai ler a informação na linha de "DATA" mais próxima, esteja ela colocada antes ou depois da linha de "READ". A instrução "GOTO" faz o MSX-BASIC voltar de novo à linha 10. O erro da mensagem "Out of DATA in 10" é causado pelo facto de não haver mais dados disponíveis no arquivo "DATA" quando a instrução "READ" foi executada pela quarta vez.

A instrução "RESTORE" força a primeira instrução "READ" seguinte a ler a informação da instrução "DATA" desde o primeiro elemento. Acrescente-se a linha seguinte ao programa anterior e corra-se o mesmo.

```
15 RESTORE
```

O programa agora repete indefinidamente "JAN" até que o parem premindo simultaneamente as teclas CTRL e STOP. Utilize o comando "NEW" para limpar a memória outra vez.

**Tabelas
com variáveis**

Também se podem armazenar variáveis numa tabela, utilizando a instrução "DIM". Exemplo:

```
10 DIM A(20)
```

Isto quer dizer que 21 áreas na memória (de 0 a 20) estão reservadas e que todas elas têm o nome "A". Quando quisermos usar uma das variáveis numa linha de

programa, é evidente que tal nome não é suficiente. Teremos de incluir um número entre parêntesis imediatamente depois do nome da variável, claro que o número se situará entre 0 e 20.

```
20 PRINT A(3)
```

Em vez dum número também se pode usar o nome de outra variável, tal como:

```
NEW
10 DIM A(18)
20 A(3)=15:B=0
30 PRINT A(B)
RUN
0
```

A resposta que o MSX-BASIC dá é 0, porque o conteúdo do campo da memória A(0) é igual a zero. Agora mudemos a linha 20 do programa como segue:

```
20 A(3)=15:B=3
```

Quando prosseguirmos com o programa de novo, MSX-BASIC dará a resposta "15" que é o conteúdo do campo da memória A(3).

Tabelas de duas dimensões

É também possível reservar tabelas de duas dimensões em MSX-BASIC, como segue:

```
10 DIM A(1,3)
```

Isto reserva a seguinte área na memória:

	0	1	2	3
0				
1				

Ao referimo-nos à variável "A" numa linha de programa, ambas as dimensões terão que ser mencionadas, como segue:

```
20 A(1,2)=64
```

Essa instrução coloca o número 64 na linha 1, coluna 2, na área reservada da memória.

	0	1	2	3
0				
1			64	

Para todas as variáveis numéricas, o MSX-BASIC reservará 11 campos da memória (0 a 10), mesmo sem a instrução "DIM".

Também se podem fazer tabelas para variáveis alfanuméricas.

As tabelas usadas num programa através da instrução "DIM" podem ser apagados durante a execução do programa, usando a instrução "ERASE". Aqui está um exemplo:

```
10 DIM A(20)
20 ERASE A
30 DIM A(25)
```

Se apagarmos a linha 20 deste exemplo, o MSX-BASIC dará um erro de mensagem, porque a tabela A foi reservada duas vezes. Ao introduzir a instrução "ERASE" na linha 20, a primeira tabela "A" é apagada antes de ser aberta a segunda tabela "A".

AKTIMADONNAK PRESS 2013

Decisões

Capítulo 2.7 A execução dum programa pode ser variada usando a introdução "IF", como no exemplo seguinte:

```
10 INPUT A,B
20 IF A=B THEN PRINT "A=B" ELSE PRINT "A<>B"
```

Aqui o conteúdo das variáveis "A" e "B" são comparados na linha 20. Com a instrução "IF", o MSX-BASIC é inquirido se a condição é verdadeira. Se é, a instrução a seguir à palavra "THEN" é executada. Se não é, seguirá as instruções identificadas pela palavra "ELSE".

A palavra "ELSE" e as instruções subsequentes da linha de programa começando com a instrução "IF" são opcionais. Isto quer dizer que podemos não as usar. Se a linha não contém a palavra "ELSE", o programa prosseguirá com a linha seguinte quando a condição não for verdadeira.

Se a palavra "THEN" é seguida pela instrução "GOTO" não é preciso acrescentar a palavra "THEN". Uma alternativa é apagar a palavra "GOTO", como ilustrado no exemplo seguinte:

```
10 INPUT A,B
20 IF A=B THEN 50 ELSE GOTO 40
30 END
40 PRINT "A<>B":GOTO 30
50 PRINT "A=B":GOTO 30
```

As seguintes condições podem ser incluídas na instrução "IF":

- < Menor que
- > Maior que
- = Igual # a
- <> Diferente
- >= Igual ou maior que
- <= igual ou menor que

Se uma instrução "IF" contém um cálculo matemático, o cálculo é executado antes de seguir para a declaração condicional.

Exemplo:

```
10 INPUT A,B
20 IF A MOD B=0 THEN PRINT "REST=0"
```


A instrução a seguir à palavra THEN só é executada se o resultado do módulo de cálculo "A MOD B" for igual a 0.

Usando a palavra "NOT", a condição pode conter uma conclusão negativa, como se mostra no exemplo seguinte:

```
10 INPUT A,B
20 IF NOT A=B THEN PRINT "A<>B"
30 IF A<>B THEN PRINT "A<>B"
```

As linhas 20 e 30 do programa conduzirão ao mesmo resultado.

Combinações de condições

É possível incluir mais que uma condição a seguir à instrução "IF". O resultado final depende da relação entre estas condições:

1. Relação dada: "AND" (E)

Condição 1	Condição 2	Resultado
Verdadeiro	Verdadeiro	Verdadeiro
Verdadeiro	Não verdadeiro	Não verdadeiro
Não verdadeiro	Verdadeiro	Não verdadeiro
Não verdadeiro	Não verdadeiro	Não verdadeiro

2. Relação dada: "OR" (OU)

Condição 1	Condição 2	Resultado
Verdadeiro	Verdadeiro	Verdadeiro
Verdadeiro	Não verdadeiro	Verdadeiro
Não verdadeiro	Verdadeiro	Verdadeiro
Não verdadeiro	Não verdadeiro	Não verdadeiro

3. Relação dada: "XOR"

Condição 1	Condição 2	Resultado
Verdadeiro	Verdadeiro	Não verdadeiro
Verdadeiro	Não verdadeiro	Verdadeiro
Não verdadeiro	Verdadeiro	Verdadeiro
Não verdadeiro	Não verdadeiro	Não verdadeiro

4. Relação dada: "EQV"

Condição 1	Condição 2	Resultado
Verdadeiro	Verdadeiro	Verdadeiro
Verdadeiro	Não verdadeiro	Não verdadeiro
Não verdadeiro	Verdadeiro	Não verdadeiro
Não verdadeiro	Não verdadeiro	Verdadeiro

5. Relação dada: "IMP"

Condição 1	Condição 2	Resultado
Verdadeiro	Verdadeiro	Verdadeiro
Verdadeiro	Não verdadeiro	Não verdadeiro
Não verdadeiro	Verdadeiro	Verdadeiro
Não verdadeiro	Não verdadeiro	Verdadeiro

Exemplos:

```
10 IF 3>1 AND 3>2 THEN PRINT "CORRECT"
20 IF 3>1 OR 6<7 THEN PRINT "CORRECT"
30 IF 3>1 XOR 6<1 THEN PRINT "CORRECT"
40 IF 4>3 EQV 6<7 THEN PRINT "CORRECT"
50 IF 4<3 IMP 3>2 THEN PRINT "CORRECT"
```

Com todas as linhas de programa acima referidas, as instruções que seguem a palavra "THEN" são executadas.

Como comparar dados alfanuméricos

Não só a informação numérica mas também os dados alfanuméricos podem ser utilizados numa condição. A comparação é então feita comparando o código do carácter de todos os símbolos usados. Encontra-se no Apêndice E deste manual um sumário de todos os códigos do carácter.

Exemplos:

```
10 IF "A"<"B" THEN PRINT "A IS SMALLER THAN B"
20 IF A$<B$ THEN PRINT "THE CONTENT OF A$ IS SMALLER
  THAN THE CONTENT OF B$"
30 IF A$="JOHN" THEN PRINT "THE CONTENT OF A$ IS JOHN"
40 IF "JOAN"<"JOHN" THEN PRINT "CORRECT"
```

Informações numéricas e alfanuméricas não podem ser combinadas numa condição simples.

"Loops" de programa

Limpe a memória do computador escrevendo a ordem "NEW" e componha depois as linhas seguintes:

```
10 CT=1
20 PRINT "MSX-BASIC"
30 CT=CT+1
40 IF CT<11 GOTO 20
RUN
```

Ao ver este programa verificamos que o texto "MSX-BASIC" aparece dez vezes no écran. Há uma maneira mais simples de conseguir o mesmo resultado:

```
NEW
10 FOR CT=1 TO 10
20 PRINT "MSX-BASIC"
30 NEXT CT
RUN
```


Obtém-se assim o mesmo resultado que com o exemplo precedente. Na linha 10 do programa ao campo de memória «CT» é dado o valor 1. Depois, o texto "MSX-BASIC" é impresso na linha de programa 20. Na linha 30 o valor de "CT" é aumentado 1. Quando "CT" é maior do que 10, a linha 40 será executada. Enquanto "CT" for mais pequeno ou igual a 10, as linhas 20 e 30 do programa serão repetidas.

No nosso exemplo com a instrução "NEXT", a variável "CT" é aumentada com o valor 1. Este valor pode ser mudado utilizando a opção "STEP" na instrução "FOR". Os valores numa instrução "FOR" podem também ser substituídos por uma variável, como no próximo exemplo:

```
NEW
10 A=4
20 FOR CT=8 TO A STEP -2
30 PRINT "MSX-BASIC"
40 NEXT CT
RUN
```

Neste exemplo, o valor da nossa variável "CT" não é aumentado com a subsequente instrução "NEXT", mas é na verdade reduzido pelo valor 2 da opção "STEP". Isto significa que o valor inicial da nossa variável "CT" tem de ser mais alto que o valor final. Ao fazer RUN ao programa acima veremos que o texto "MSX-BASIC" é reproduzido três vezes.

Agrupamento O exemplo seguinte é um programa para agrupamento de uma selecção de números dados pelo utilizador e para os colocar em sequência. O utilizador tem de dar a conhecer ao computador quantos números estão envolvidos e depois escreve os números. Os números são armazenados numa tabela e agrupados em sequência de acordo com o método binário.

```
NEW
10 DIM G(100)
20 INPUT "QUE NUMEROS PRETENDE (ENTRE 2 E 100)";A
30 IF A<2 OR A>100 GOTO 20
40 FOR I=1 TO A
50 PRINT "NUMBER"; I;
60 INPUT G(I)
70 NEXT I
75 REM QUANDO I SEJA MAIS PEQUENO QUE
O CONTEUDO DE A, A EXECUCAO CONTINUA NA LINHA 50"
80 PRINT "ESTOU A CLASSIFICAR "
90 FOR I=2 TO A
100 X1=1:X2=I
110 X3=X1+INT((X2-X1)/2)
120 IF G(I)<G(X3) THEN 150
130 IF G(I)>G(X3) THEN 170
140 X2=X3
150 IF X2=X3 THEN 190
160 X2=X3:GOTO 110
170 IF X1=X3 THEN 190
180 X1=X3 :GOTO 110
190 H=G(I)
200 FOR J=I TO X2+1 STEP -1
210 G(J)=G(J-1)
220 NEXT J
230 G(X2)=H
```



```

240 NEXT I
245 REM QUANDO I SEJA MENOR QUE O CONTEUDO DE (A)
A EXECUCAO CONTINUA NA LINHA 100
250 PRINT "EIS OS NUMEROS ORDENADOS"
260 FOR I=1 TO A
270 PRINT G(I);
280 NEXT I

```

Ao ler este programa veremos que os números estão correctamente colocados em sequência. Se o programa não funcionar devidamente, verificar cuidadosamente se o mesmo foi correctamente escrito.

Programa com "escolha"

Programas capazes de executar várias funções, começam muitas vezes com uma "ementa à escolha" (choice menu). O menu mostra um index das secções de programa e essa secção particular do programa é executada quando o número apropriado for escolhido. Aqui vai um exemplo dum programa menu:

```

NEW
10 PRINT "CHOICE MENU"
20 PRINT "1=PROGRAM LINE 100"
30 PRINT "2=PROGRAM LINE 200"
40 PRINT "3=END PROGRAM"
50 INPUT "YOUR CHOICE";A
60 IF (A<1)OR(A>3) GOTO 50
70 ON A GOTO 100,200,300
100 PRINT "THIS IS PROGRAM LINE 100":GOTO 10
200 PRINT "THIS IS PROGRAM LINE 200":GOTO 10
300 END
RUN

```

A linha mais interessante deste programa é a n.º 70. Quando o conteúdo da variável "A" é igual a 1, o programa continua com a linha 100. Se é igual a 2, o programa continuará com a linha 200 e com o valor 3, o programa executa a linha de programa 300.

Tratamento de erros

O MSX-BASIC torna possível localizar e tratar erros que ocorrem durante a execução dum programa. Aqui está um exemplo:

```

NEW
10 ON ERROR GOTO 50
20 INPUT "NUMBER";A%
30 END
50 IF ERR=6 THEN PRINT "THE NUMBER SHOULD BE A NUMBER
BETWEEN -32768 and +32767":RESUME 0
60 ON ERROR GOTO 0
RUN

```

Neste exemplo a instrução da linha 10 faz com que o programa continue na linha 50 se um erro for descoberto. Se o utilizador escrever um número menor que - 32768 ou maior que + 32767 em resposta à pergunta na linha 20, o MSX-BASIC indicará um erro de excesso (overflow) porque a variável "A%" é uma variável de expressão dum número inteiro.

O erro de excesso (overflow) tem o código de erro 6 (ver Apêndice A deste manual). A linha 50 do programa pergunta se o código de erro 6 é de facto a causa do erro de rotina. Se assim for, o texto da linha 50 é mostrado no écran.

Depois disso a instrução "RESUME" é utilizada para continuar o programa, começando na linha 20. Com a linha 60 o MSX-BASIC quebra a execução do programa se o erro da rotina for causado por outro erro do que o coberto pelo código 6. O MSX-BASIC também torna possível ao utilizador definir códigos de erro. Só há que lembrar que os códigos de erro que utilizarmos devem ser mais altos que os usados pelo MSX-BASIC. Aqui temos um exemplo:

```
NEW
10 ON ERROR GOTO 100
20 INPUT "END PROGRAM";A$
30 IF A$="YES" THEN ERROR 250
40 PRINT "PROGRAM EXECUTION":GOTO 20
50 END
100 IF ERR=250 THEN INPUT "ARE YOU SURE";A$:IF
A$="YES" THEN RESUME 50 ELSE RESUME 20
110 ON ERROR GOTO 0
RUN
```

Na linha 30 do programa, o código de erro 250 é usado quando a variável "A\$" é idêntica a "YES" (sim). Nesse caso, a linha 100 será executada como indicado na linha 10. A linha 100 pergunta se o erro de rotina é causado pelo código de erro 250. Se assim for, o MSX-BASIC pedirá a confirmação ao utilizador. Se a resposta do utilizador for positiva, o programa continua utilizando a instrução "RESUME 50", com a linha 50. Se a resposta do utilizador for negativa, o programa continuará com a linha 20, utilizando a instrução "RESUME 20".

Sub-rotinas

Capítulo 2.8 Vamos assumir que o programa seguinte está armazenado na memória do computador:

```
10 INPUT A:INPUT B
20 C=A*100/B:PRINT A;
30 PRINT "IS";C;"PERCENT OF";
40 PRINT B
50 C=B*100/A:PRINT B;
60 PRINT "IS";C;"PERCENT OF";
70 PRINT A
80 END
```

Notaram, sem dúvida, que a linha 60 do programa é idêntica à linha 30. Podem também utilizar uma linha do programa e referir o computador para esta linha quando ela chega à linha 60 com a instrução "GOSUB".

O nosso exemplo terá então o seguinte aspecto:

```
10 INPUT A:INPUT B
20 C=A*100/B:PRINT A;
30 GOSUB 90
40 PRINT B
50 C=B*100/A:PRINT B;
60 GOSUB 90
70 PRINT A
80 END
90 PRINT "IS";C;"PERCENT OF";
100 RETURN
```

Ao chegar à linha 30, o MSX-BASIC continuará o programa na linha 90. A instrução "RETUR" faz-se o MSX-BASIC voltar para a linha 40. O mesmo acontece quando chega à linha 60. Quando chega à linha 100, no entanto, o MSX-BASIC volta para a linha 70. As instruções das linhas 90 e 100 do programa chamam-se "sub-rotinas". Quando o MSX-BASIC chega à linha 90 como resultado da instrução "GOSUB", lembra-se de onde veio e volta atrás quando chega à instrução "RETURN".

A linha 80 do programa tem particular importância no nosso exemplo. Se o esquecer, e se a linha 90 se seguir imediatamente à linha 70, o MSX-BASIC daria o erro "RETURN WITHOUT GOSUB" ao chegar à linha 100.

**Sub-rotinas
a pedido**

Uma sub-rotina pode também ser chamada com a indicação de:

1. O conteúdo duma variável
2. A hora
3. Uma ou mais teclas de funções
4. Teclas STOP e CTRL
5. Colisão entre dois "sprites"
6. O botão de acção do comando manual

As duas últimas indicações são tratadas com mais pormenor nos capítulos 2.12 e 2.14. Uma sub-rotina é chamada com a indicação do conteúdo duma variável utilizando a palavra "ON". Aqui está um exemplo:

```
10 INPUT A
20 IF (A<0)OR(A>255) GOTO 10
30 ON A GOSUB 100,200,300
40 END
100 PRINT "SUBROUTINE 100":RETURN
200 PRINT "SUBROUTINE 200":RETURN
300 PRINT "SUBROUTINE 300":RETURN
```

Neste exemplo, a linha de sub-rotina 100 é executada se "A" for igual a 1. A linha de sub-rotina 200 é executada quando "A" = 2. O conteúdo da variável "A" na linha 30 não pode ser negativo ou maior que 255. Se for, o MSX-BASIC dará um erro de mensagem. Em vez da variável "A" uma forma matemática pode também ser dada, tal como:

```
30 ON A+1 GOSUB 100,200,300
```

Quando o conteúdo da variável "A" é 1, a linha 200 da sub-rotina será executada. Quando se trata de cálculos, a mesma regra se aplica, isto é, o resultado não pode ser negativo nem maior que 255. Quando a variável "A" ou o resultado do cálculo é maior que o número de linhas "GOSUB", a instrução seguinte é executada. Se, por exemplo, a nossa variável "A" = 3, então a linha 40 será executada porque "A" + 1 = 4 e só três linhas de sub-rotinas entraram, seguindo "GOSUB" na linha 30.

Se uma sub-rotina é chamada com a indicação de tempo, teclas de funções e a tecla STOP juntamente com a tecla CTRL, estas funções têm que ser activadas quando necessário.

```
10 ON STOP GOSUB 100
20 ON INTERVAL=250 GOSUB 200
30 ON KEY GOSUB 300
40 STOP ON:INTERVAL ON:KEY(1) ON
50 GOTO 50
100 PRINT "END":END
200 PRINT "ANOTHER 5 SECONDS GONE BY"
210 RETURN
300 PRINT "FUNCTION KEY 1 PRESSED IN"
310 RETURN
```

No exemplo acima, as linhas de programa 10, 20 e 30 indicam as sub-rotinas para onde o programa tem de saltar quando as teclas CTRL e STOP são premidas simultaneamente (linha 10) quando o tempo tiver expirado (neste caso na linha 20

com 5 segundos, sendo $250 \times 1/50$ seg.) e quando a tecla de função 1 é premida. Na linha 40 do programa, as funções pedidas nas linhas 10 a 30 são activadas. A linha 50 do programa parece ser absolutamente desnecessária já que quer dizer que o MSX-BASIC continuará a executar a linha de programa 50 indefinidamente!

Enquanto faz isso, no entanto, o MSX-BASIC verificará se uma das teclas activadas foi premida ou se o intervalo pedido expirou. Nesse caso, as respectivas sub-rotinas serão executadas. Ter especial atenção ao facto de que a linha 100 da sub-rotina não está terminada com "RETURN", mas com uma instrução "END". De outro modo seria impossível parar o programa sem desligar o computador!

Qual o dia? No nosso próximo exemplo, o MSX-BASIC determinará o dia da semana quando introduzir uma data específica. Primeiro limpe a memória com o comando "NEW". Depois escreva o programa seguinte:

```
2 KEYOFF
10 DATA DOMINGO,SEGUNDA,TERCA,QUARTA,QUINTA,SEXTA,SABADO
20 PRINT "DE QUE DATA SE TRATA";PRINT
30 INPUT "DIA (1 A 31)";DD
40 IF DD<1 OR DD>31 GOTO 30
50 INPUT "MES (1-12)";MM;PRINT
60 IF MM<1 OR MM>12 GOTO 50
62 IF MM<3 AND DD>29 GOTO 30
64 IF DD>30 GOTO 200
70 INPUT "ANO (1901-2099)";AA;PRINT
80 IF AA<1901 OR AA>2099 GOTO 70
84 IF INT(AA/4)*4=AA AND MM<3 AND DD>28 GOTO 300
90 IF MM>2 GOTO 110
100 AA=AA-1; MM=MM+12
110 H1=INT(AA/100);H2=AA-H1*100
120 H3=INT(2.6001*(MM-2)-.2)+DD+H2+INT(H2/4)+INT(H1/4)-2*H1
130 S1=H3-INT(H3/7)*7+1
132 REM .....
140 RESTORE
150 FOR I=1 TO S1:READ H$:NEXT I
156 IF MM>12 THEN MM=MM-12;AA=AA+1
160 PRINT DD;MM;AA;" E ";H$
170 K$=INKEY$;IF K$="" GOTO 170
180 CLS;GOTO 10
200 IF MM=4 OR MM=6 OR MM=9 OR MM=11 THEN PRINT "MES C/ 30 DIAS";
GOTO 30
202 GOTO 70
```

Execute este programa e escreva a data como pedido pelo MSX-BASIC. Se o programa não funcionar devidamente, verifique cuidadosamente se foi cometido algum erro ao dar entrada das várias instruções!

Funções

Capítulo 2.9 Em complemento de ordens e instruções, o MSX-BASIC tem também um largo número de funções. Estas funções dão o resultado de um cálculo com uma constante ou uma variável. Estas funções podem apenas ser usadas em combinação com uma instrução. Aqui está um exemplo:

```
NEW
10 PRINT SIN(5)
RUN
-.9589242746631
```

ou:

```
NEW
10 X=5
20 PRINT SIN(X)
RUN
-.9589242746631
```

No Manual de Referência encontra-se um sumário completo das funções disponíveis. Podem ser divididas em vários grupos:

1. Funções matemáticas

"ABS", "ATN", "COS", "EXP", "INT", "LOG", "SGN", "SIN", "SQR", "TAN".

2. Funções alfanuméricas

"ASC", "LEFT\$", "LEN", "MID\$", "RIGHT\$", "STR\$", "STRING\$", «VAL», "BIN\$", "HEX\$", "OCT\$", "INSTR\$".

3. Funções de conversão

"CDBL", "CINT", "CSNG", "FIX".

4. Funções input

"INKEY\$", "INPUT\$".

5. Funções output

"CHR\$", "POS", "LPOS", "SPC", "TAB".

6. Funções diversas

"CSLRIN", "ERL", "ERR", "FRE", "PEEK", "RND", "USR", "VARPTR", "POINT", "VPEEK", "STICK", "STRIG", "PDL", "PAD", "PLAY", "EOF".

O utilizador pode também definir as suas próprias funções usando a instrução "DEF FN" (ver capítulo 2.4)

Número ao acaso A função "RND" é por vezes usada para fazer o MSX-BASIC determinar um número ao acaso. Este número será entre 0 e 1. Se o utilizador quiser um número ao acaso entre 1 e 6 (para um dado, por exemplo) terá que multiplicar o número por 6 e juntar-lhe 1. Aqui temos um exemplo:

```
NEW
10 A=INT(RND(1)*6+1)
20 PRINT A
RUN
```

Números de Loto No exemplo acima usaram-se duas funções em combinação com uma instrução simples na linha 10. A função "INT" faz do número seleccionado ao acaso uma expressão de número inteiro.

O próximo exemplo é um programa que escolhe 6 números de loto:

```
NEW
10 X=0
20 FOR I=1 TO 6
30 A=INT(RND(1)*41+1)
50 IF A=X(1) GOTO 30
60 IF A=X(2) GOTO 30
70 IF A=X(3) GOTO 30
80 IF A=X(4) GOTO 30
90 IF A=X(5) GOTO 30
110 X(I)=A
120 NEXT I
130 PRINT "THE LOTTO NUMBERS ARE";
140 FOR I=1 TO 6:PRINT X(I);:NEXT I:PRINT
160 INPUT "ONCE MORE (Y/N)";AS
170 IF AS="Y" GOTO 10
180 END
```

Veja o programa e descubra que números de loto o MSX-BASIC lhe apresenta. Se o programa não funcionar devidamente verifique cuidadosamente se todas as instruções foram devidamente introduzidas.

Espaço de memória A função "FRE" permite saber quanto espaço de memória está ainda disponível. Tentemos isto:

```
PRINT FRE(0)
nnnnn
```

Isto significa que ainda temos nnnnn bytes livres para MSX-BASIC.

Se se pretende saber que espaço da memória existe para variáveis alfanuméricas, use-se esta instrução:

```
PRINT FRE("")
xxx
```


Isto quer dizer que ainda temos “xxx” bytes livres para variáveis alfanuméricas.

AKTIMADONNAK PRESS 2013

O MSX-BASIC e o écran

Capítulo 2.10 Com o MSX-BASIC o écran pode ser usado de quatro maneiras diferentes:

1. Modo de texto 1
2. Modo de texto 2
3. Modo gráfico 1
4. Modo gráfico 2

Modo de texto 1 Neste modo um máximo de 40 caracteres pode ser colocado numa linha de écran, que tem 24 linhas. Cada posição no écran é definida pela coordenada X (indicando a coluna) e a coordenada Y (indicando a linha). A mais pequena coordenada X é o 0 (zero) e representa a coluna mais à esquerda, enquanto que a coluna 39 é a mais alta e representa a coluna mais à direita. A coordenada Y vai de 0 (zero) para a linha de cima, a 23 para a linha de baixo. Com a instrução "WIDTH" pode dizer-se ao computador quantos caracteres queremos por linha. Com a instrução "LOCATE" o cursor pode ser deslocado para qualquer posição desejada no écran. Este modo de texto (modo de texto 1) pode ser activado com a instrução "SCREEN 0". Aqui temos um exemplo:

```
NEW
10 SCREEN 0
20 WIDTH 40
30 LOCATE 15,10:PRINT "HELLO"
RUN
```

Neste exemplo, o modo de texto 1 é activado na linha 10. A linha 20 abre a possibilidade de pôr 40 caracteres numa linha. Na linha 30 o cursor move-se para a posição 15 (16.^a coluna) da linha 10 (11.^a linha) onde a palavra "HELLO" está impressa. Quando o MSX-BASIC está ao nível de comando, ou se tem de processar uma instrução "INPUT", o écran está sempre em modo de texto.

Modo de texto 2 Este modo de texto permite um máximo de 32 caracteres por linha no écran. Cada posição no écran é de novo definida pela coordenada X para a coluna e a coordenada Y para a linha. A coordenada X é um número entre 0 (zero) para a coluna esquerda e 31 para a coluna direita. A coordenada Y vai de 0 (zero) para a linha de cima a 23 para a linha de baixo. O número máximo de caracteres por linha pode ser determinado usando a instrução "WIDTH".

Utilizando a instrução "LOCATE" o cursor pode ser deslocado para qualquer posição desejada no écran.
Este modo de texto (modo de texto 2) é activado com a instrução "SCREEN 1".
Aqui está um exemplo:

```
NEW
10 SCREEN 1
20 WIDTH 32
30 LOCATE 15,10:PRINT "HELLO"
RUN
```

Este exemplo conduz exactamente ao mesmo resultado do exemplo precedente, descrevendo o modo de texto 1.
Na linha 10 do programa, no entanto, activámos agora o modo de texto 2.
Na linha 20 criámos a possibilidade de usar 32 caracteres por linha. Se tivermos dificuldades em ler os caracteres no modo de texto 1, utilizamos o modo de texto 2 de MSX-BASIC.

Há áreas de cor azul ciânico no topo e no fundo do écran no modo de texto 2.
Podem mudar-se as cores destas áreas utilizando a instrução "COLOR".

Para voltar de novo ao modo de texto 1, escreve-se:

```
SCREEN 0:WIDTH 40
```

Modo gráfico 1 Neste modo o écran é dividido em 256 pontos horizontalmente e em 192 pontos verticalmente. A posição de cada ponto é determinada por uma coordenada X (a coluna) e por uma coordenada Y (a linha).
A coordenada X pode ser qualquer número entre 0 (zero) para a coluna da esquerda, a 255 para a coluna da direita. A coordenada Y pode ir de 0 (zero) para a linha do topo a 191 para a linha de baixo.
Neste modo apenas instruções gráficas podem ser usadas para criar uma imagem no écran. É activado pela instrução "SCREEN 2". As instruções gráficas são explicadas no capítulo 2.11.
Exemplo:

```
NEW
10 SCREEN 2
20 CIRCLE (185,115),10
30 GOTO 30
RUN
```

Neste exemplo o computador desenhará um círculo no écran, usando o ponto 185 (a coordenada X) e 115 (a coordenada Y) como centro.
A linha 30 tem aqui particular importância. Utilizando a instrução "GOTO" a linha 30 do programa é executada repetidamente. Se a linha 30 não tivesse sido acrescentada, o MSX-BASIC voltaria ao nível de comando depois de executadas as linhas 10 e 20. O modo de texto 1 seria então activado automaticamente. Considerando a velocidade com que o MSX-BASIC opera, nem teríamos tido tempo de ver o círculo.
Pode parar-se o programa premindo simultaneamente as teclas CTRL e STOP.

Modo gráfico 2 Neste modo o écran está dividido exactamente da mesma maneira como no modo gráfico 1. A diferença entre os dois é que as imagens que vão aparecendo no écran são todas construídas a partir de pequenos quadrados medindo 4 x 4 pontos de imagem.

Quando o modo gráfico é activado com a instrução "SCREEN 3" — trabalhando deste modo apenas instruções gráficas podem ser usadas. Encontram-se mais pormenores sobre as várias instruções gráficas no capítulo 2.11. Vejamos um exemplo:

```
NEW
10 SCREEN 3
20 LINE (125,10)-(200,85)
30 GOTO 30
RUN
```

Neste exercício uma linha será desenhada do ponto 125 (coordenada X), 10 (coordenada Y) ao ponto 200 (coordenada X), 85 (coordenada Y). A linha é feita de pequenos quadrados. Pode parar-se o programa premindo simultaneamente as teclas CTRL e STOP. Vejamos o que acontece ao mudar a linha 10 para:

```
10 SCREEN 2
```

e desenrole o programa novamente.

Quando tivermos visto a diferença entre o modo gráfico 1 e 2, pára-se este exemplo premindo simultaneamente as teclas CTRL e STOP.

**Como limpar
o écran**

Pode limpar-se o écran com a instrução "CLS", que se usa com todos os modos do écran.

Instruções para cor e gráficos

Capítulo 2.11 O MSX-BASIC permite o uso de 16 cores. Estas cores são numeradas de 0 a 15. Podem ser usadas como cores de primeiro plano ou de fundo e para cores do bordo, no modo texto 2 ou modos gráficos 1 e 2. Uma lista completa das várias cores é dada no Apêndice C, com os seus números de código.

A cor registada sob o número de cor 0 (transparente) tem um interesse especial. Quando este número de cor é usado, as cores em evidência apenas aparecerão nas áreas cobertas pela cor transparente. O que significa que a cor número 0 é apenas usada quando é necessária como cor de primeiro plano. Quando ambas as cores quer de primeiro plano quer de fundo são transparentes, a área aparecerá em preto.

As cores podem ser escolhidas utilizando a instrução "COLOR". Limpe-se a memória do computador com a ordem "NEW" e escreva-se o seguinte exemplo:

```
10 SCREEN 1
20 COLOR 1,3,6
30 PRINT "HELLO"
RUN
```

Neste exemplo as letras são pretas como cor de primeiro plano (cor número 1) contra cor de fundo verde claro (cor número 3). As áreas limites são vermelho escuro (cor número 6).

Depois de vermos este exemplo, escreveremos:

```
SCREEN 0:COLOR 15,4,7
```

Quando se começa a trabalhar com o MSX-BASIC, as cores standard no modo de texto 1 são brancas no primeiro plano (cor número 15) e azul escuro na cor de fundo (cor número 4).

Nos outros modos de écran veremos também as áreas limites em azul ciânico (cor número 7) como cor standard.

Como desenhar uma linha

Com a instrução "LINE" pode desenhar-se uma linha no écran entre dois pontos. Estes pontos devem estar indicados com as coordenadas X e Y com um valor relativo ou absoluto. Um valor relativo é indicado pela palavra "STEP" quando as coordenadas são indicadas. Um ponto na coluna 11, linha 16, por exemplo, tem coordenadas absolutas (10,15).

Vamos supor que a última coordenada absoluta X-, e Y que foram usadas são (3,6), então o ponto na coluna 11, 16.^a coordenada de linha (10,15) podem também ser indicadas como STEP (7,9).

```
NEW
10 SCREEN 2
20 LINE (125,10)-STEP(75,75),1
30 GOTO 30
RUN
```

Neste exemplo as coordenadas do segundo ponto são dadas em relação ao primeiro ponto. O "1" que está no fim da linha de programa 20 determina a cor da linha. Preto, neste caso (cor número 1). Isto significa que a linha 20 do programa podia também ter sido escrita como segue:

```
20 LINE (125,10)-(200,85),1
```

Juntando a letra "B" depois do número da cor na instrução "LINE", MSX-BASIC produzirá um retângulo no écran com a linha em diagonal. A própria linha não aparecerá mais. Páre-se o programa premindo simultaneamente as teclas STOP e CTRL. Faça-se agora a seguinte alteração no programa:

```
20 LINE (125,10)-STEP(75,75),1,B
```

e corra o programa novamente.

Se as letras "BF" forem escritas depois do número da cor na instrução "LINE", o MSX-BASIC desenhará um rectângulo no écran e preencherá a área com a cor, de acordo com o número de cor dado na instrução "LINE". Páre-se o programa premindo simultaneamente as teclas CTRL e STOP e mude-se a linha 20 para:

```
20 LINE (125,10)-STEP(75,75),1,BF
```

Corra o programa de novo. Para parar premir simultaneamente as teclas CTRL e STOP.

Como desenhar um círculo

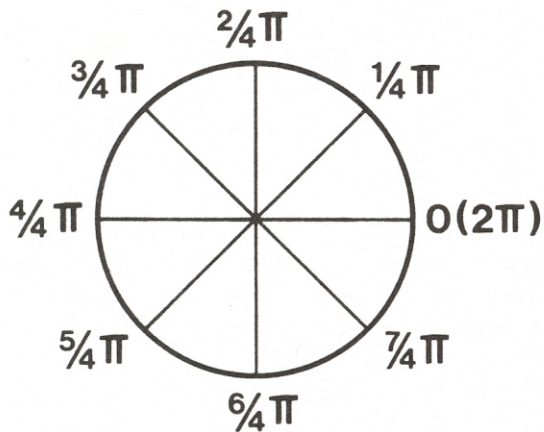
Com a instrução "CIRCLE" pode desenhar-se um círculo no écran a partir dum ponto específico, determinado pelas coordenadas X e Y. Estas coordenadas podem ter um valor absoluto ou relativo. O raio é dado em termos de pontos de imagem. Aqui temos um exemplo:

```
NEW
10 SCREEN 2
20 CIRCLE (90,80),20,1
30 GOTO 30
RUN
```

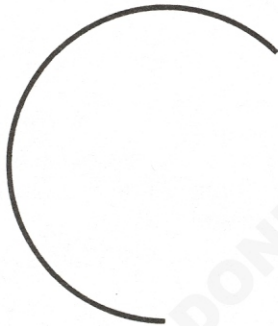
Neste exemplo é desenhado um círculo preto (cor número 1) à volta do ponto central 90 (coordenada X), 80 (coordenada Y) com um raio de 20 pixel (pontos de imagem) (isto significa obviamente que o diâmetro é de 40 pixel). Com a instrução "CIRCLE" pode também pedir-se um ângulo no começo e no fim.

Ambos são determinados de acordo com a seguinte ilustração:

$$\pi = 3.1415926535898$$



vamos supor que queremos desenhar parte de um círculo como é indicado na ilustração seguinte:



Primeiro pára-se o programa do exemplo precedente premindo simultaneamente as teclas CTRL e STOP. Depois muda-se a linha 20 como segue:

```
20 CIRCLE (90,80),20,1,0.7854,4.7124
```

Corra o programa novamente. O ponto de partida é calculado como segue: $3.1416 \times 1/4 = 0.7854$ e o ponto de término como $3.1416 \times 6/4 = 4.7124$. Os pontos inicial e final do círculo podem ser ligados com o ponto central introduzindo valores negativos para ambos os pontos. Pára-se o programa primeiro (CTRL e STOP) e muda-se a linha 20 para:

```
20 CIRCLE (90,80),20,1,-0.7854,-4.7124
```

É também possível, claro, ligar apenas um dos pontos com o ponto central. O MSX-BASIC desenhara também uma elipse no écran. Para isto introduza-se um número indicando a proporção entre o raio horizontal e o vertical da elipse. Primeiro pára-se o programa anterior e muda-se a linha 20 para:

```
20 CIRCLE (90,80),20,1,,,2
```

Certifiquemo-nos de que escrevemos todas as vírgulas antes de recomeçar o programa. Se o não fizermos, o MSX-BASIC tomará o "2", que foi introduzido para indicar a proporção entre ambos os sinais, como ponto de partida do círculo. Neste exemplo o raio vertical é 20 e o horizontal $20:2 = 10$.

Páre-se de novo o programa depois de se ter experimentado isto (teclas CTRL e STOP).

Como pintar Com o MSX-BASIC é possível colorir uma área definida por linhas. Para fazer isto é preciso usar a instrução "PAINT". Com esta instrução a posição tem de ser indicada por meio das coordenadas X e Y e o número da cor. Estas coordenadas podem representar quer um valor absoluto quer um valor relativo. Do ponto de partida dado, o MSX-BASIC procurará uma área dentro das linhas da mesma cor que a indicada pela instrução "PAINT". Preencherá então a área com essa cor. Tentemos este exemplo:

```
NEW
10 SCREEN 2
20 CIRCLE (50,80),30,1
30 CIRCLE (90,80),30,1
40 PAINT (70,80),1
50 GOTO 50
RUN
```

Neste exemplo o MSX-BASIC desenha dois círculos pretos no écran que se sobrepõem parcialmente. Com a instrução "PAINT" o cursor é colocado na área onde os círculos se sobrepõem, que é pintada a preto (cor 1 na linha 40). Páre-se de novo o programa e mude-se a linha 20 para:

```
20 CIRCLE (50,80),30,6
```

Corra o programa de novo. A instrução «PAINT» preenche agora o círculo da linha 30 completamente com preto, uma vez que é a única área dentro das linhas pretas. Note-se que o ponto de intersecção de duas linhas tem sempre a cor da última instrução gráfica!

No nosso exemplo, o círculo vermelho da linha 20 já não forma uma linha contínua. Preencher este círculo com uma cor especial terá agora consequências tremendas. Primeiro páre-se o programa e mude-se a linha 40 para:

```
40 PAINT (70,80),6
```

Corra o programa novamente. O MSX-BASIC encherá agora todo o écran com vermelho, uma vez que não existe uma área completamente fechada por linhas vermelhas. Os pontos de intersecção de ambos os círculos formam "derrames". Páre-se de novo o programa (CTRL e STOP).

No modo gráfico 2 o MSX-BASIC procurará uma cor de borda como no exemplo seguinte:

```
NEW
10 SCREEN 3
20 LINE (125,10)-STEP(75,75),15,B
30 PAINT (150,15),6,15
40 GOTO 40
RUN
```


Aqui se vê que o quadrado, formado por uma linha branca (da linha 20) está pintado de vermelho pela instrução da linha 30. Ao trabalhar com o modo gráfico a instrução "PAINT" pode diferir da cor das linhas que fecham a área. Páre-se o programa (CTRL + STOP).

Como colocar um ponto

A instrução "PSET" deve ser usada para fazer uma imagem de ponto único, numa certa cor. A instrução "PRESET" é usada para a fazer desaparecer, como se pode verificar neste exemplo:

```
NEW
10 SCREEN 2
20 CIRCLE (80,80),20,1
30 FOR I=1 TO 1000
40 PSET (80,80),15
50 PRESET (80,80)
60 NEXT I
RUN
```

Neste exemplo, o ponto central do círculo (linha 20) é indicado por ponto de imagem cintilante. Trabalhando com o modo gráfico 2 tal ponto consistiria numa imagem de ponto de 4×4 .

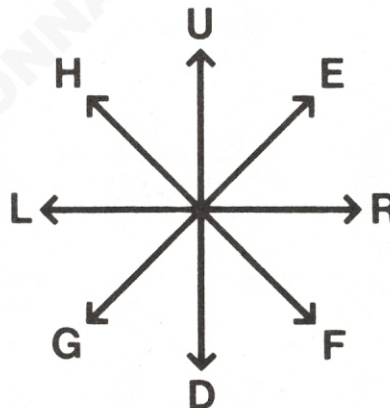
Como desenhar

Pode-se desenhar no écran com o MSX-BASIC utilizando a instrução "DRAW". Tem vários subcomandos:

1. Subcomando para desenhar uma linha

Começando na posição do cursor, pode desenhar-se uma linha em oito direcções diferentes:

Un = Norte
Dn = Sul
Ln = Oeste
Rn = Este
En = Nordeste
Fn = Sudeste
Gn = Sudoeste
Hn = Noroeste



O "n" nestes subcomandos determina o comprimento da linha em termos de pontos de imagem. Exemplo:

```
NEW
10 SCREEN 2
20 PSET (80,80),15
30 DRAW "E50"
40 GOTO 40
RUN
```

Verão que o MSX-BASIC desenha uma linha direita, diagonalmente, na direcção Nordeste. É de notar que a instrução "DRAW" é considerado valor alfanumérico, o que quer dizer que temos que usar aspas duplas.

Para desenhar uma linha desde o cursor a outra posição no écran, temos que usar a subcomando "MXY", X e Y sendo as coordenadas do ponto até ao qual a linha tem de ser desenhada. Estas coordenadas X e Y podem ter um valor absoluto ou relativo. Se forem usados valores relativos, devem ser precedidos por um mais (+) ou menos (-). Os dois próximos exemplos têm o mesmo efeito. Primeiro pare-se o programa corrente (CTRL+STOP) e escreva-se:

```
NEW
10 SCREEN 2
20 PSET (80,80),15
30 DRAW "M90,75"
40 GOTO 40
RUN
```

Páre o programa e mude a linha 30 como se segue:

```
30 DRAW "M+10,-5"
```

Passe o programa novamente.

Os subcomandos acima mencionados podem ser precedidos pelos subcomandos N e B. Se usarmos o subcomando N, o cursor voltará à sua posição inicial depois de executada a instrução "DRAW".

Páre-se o programa (CTRL+STOP) e escreva-se primeiro o seguinte exemplo:

```
NEW
10 SCREEN 2
20 PSET (80,80),15
30 DRAW "R10D10"
40 GOTO 40
RUN
```

Neste exemplo o MSX-BASIC desenha primeiro uma linha para a direita (R10) e a seguir desde esse ponto a linha descendente (D10).

Páre-se o programa e mude-se a linha 30 como segue:

```
30 DRAW "NR10D10"
```

Passe-se o programa novamente e veremos que desta vez o computador traça primeiro uma linha para a direita (R10), após o que o cursor volta para a sua posição de partida antes de desenhar uma segunda linha para baixo (D10). Se usarmos a subordem "B", o cursor é deslocado para a posição desejada, mas nenhuma linha será desenhada. Páre-se o programa e escreva-se:

```
NEW
10 SCREEN 2
20 DRAW "BM80,80R10D10"
30 GOTO 30
RUN
```


2. Subcomando para a cor

Com o subcomando "Cn" pode indicar-se a cor na qual o desenho vai ser feito. A cor é indicada pelo "n", que é o número da cor conforme lista no Apêndice C.

3. Subcomando para o ângulo do desenho

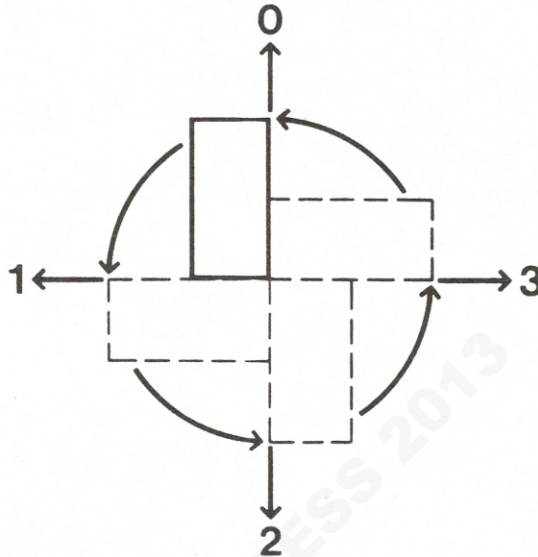
Com o subcomando "An" pode decidir-se o ângulo no qual o desenho vai ser feito, enquanto "n" é um número entre 0 e 3.

0 = 0 graus

1 = 90 graus

2 = 180 graus

3 = 270 graus



4. Subcomando para escada

O subcomando "Sn" é usado para instruir o computador quer para alargar quer para reduzir o tamanho do desenho. Aqui "n" é um número entre 0 e 255. O factor de escala é dividido por 4. O tamanho ou comprimento determinado pelas sub-ordens "U", "D", "L", "R", "E", "F", "G", "H" e a sub-ordem relativa "M" é o tamanho originalmente programado, multiplicado pelo factor de escala. A instrução: draw "S8R10", por exemplo, produzirá uma linha para a direita, não com um comprimento de 10 pontos de imagem, como instruído pela sub-ordem "R10", mas com um comprimento de 20 pontos de imagem ($10 \times 8 : 4 = 20$), como resultado do subcomando "S8".

5. Execução de subcomandos

Com o MSX-BASIC todos os subcomandos podem ser colocados numa variável alfanumérica, para que se possa usar a variável em combinação com a instrução "DRAW". Páre-se o programa precedente (CTRL + STOP) e escreva-se o seguinte exemplo:

```
NEW
10 SCREEN 2
20 A$="E20R20G20L20"
30 DRAW "AOS4BM80,80XA$;"
40 GOTO 40
RUN
```

Este programa produzirá a figura de um rombo no vosso écran. Os subcomandos necessários podem ser encontrados na variável "A\$", que é executada pela instrução "DRAW", usando a o subcomando "X". Um ponto e vírgula (;) deve ser colocado atrás da variável.

Todas as constantes usadas nos subcomandos da instrução "DRAW", podem ser substituídas por variáveis, usando a fórmula seguinte: "nome da variável;"

Por exemplo: $X1 = 10$; $X2 = 5$

DRAW "M+ = X1;,- = X2;"

A instrução "DRAW" tem muitas possibilidades. Tentemos todas. Aqui temos outro exemplo. Páre-se o programa corrente (CTRL+STOP) e escreva-se:

```
NEW
10 SCREEN 2:COLOR 15,4,4
20 PSET (127,95),1
30 FOR I=1 TO 189 STEP 4
40 A$="L"+STR$(I)+"D"+STR$(I+1)+"R"+STR$(I+2)+"U"+
  STR$(I+3)
50 DRAW "S4XA$;"
60 NEXT I
70 CLS:GOTO 20
RUN
```

O programa pode ser parado de novo premindo simultaneamente as teclas CTRL e STOP.

Sprites

Capítulo 2.12 O MSX-BASIC possibilita a criação de figuras móveis no écran. Estas figuras são conhecidas por "sprites". Uma das características típicas dos "sprites" é parecer que se movem mais ou menos em frente do écran. Por exemplo a ilustração de fundo feita com as instruções gráficas no capítulo precedente permanece intacta.

Para usar "sprites" há que obedecer a duas regras:

1. O "sprite" tem que ser definido, ou por outras palavras, temos que determinar que figura parece.
2. Há que decidir em que parte do écran aparecerá o "sprite".

Como definir os "sprites"

Os "sprites" podem ser definidos em pequeno ou grande formato. O pequeno formato significa que o "sprite" consiste em 8×8 pixel. Usando pequenos "sprites" um número total de 256 "sprites" pode ser definido. A definição de um "sprite" está na variável "SPRITE\$".

Vamos supor que queremos definir o seguinte "sprite":

8	4	2	1	8	4	2	1	
								H00
								H00
								H00
								H6C
								H92
								H10
								H00
								H00

Ver também Apêndice E

Cada 8 pontos de imagem horizontais formam 1 byte, que é de novo dividido em 2×4 bites. Estes são definidos em forma hexadecimal, como indicado à direita da nossa ilustração. O tamanho do "sprite" está indicado na instrução "SCREEN", segundo o modo de écran.

O "sprite" acima indicado pode ser codificado como segue:

```
10 SCREEN 2,0
20 A$=CHR$(&H00)+CHR$(&H00)+CHR$(&H00)+CHR$(&H6C)+
CHR$(&H92)+CHR$(&H10)+CHR$(&H00)+CHR$(&H00)
30 SPRITE$(1)=A$
```

[illegible]

```

10 SCREEN 2,2
20 A$=CHR$(&H01)+CHR$(&H03)+CHR$(&H07)+CHR$(&H07)+
CHR$(&H07)+CHR$(&H07)+CHR$(&H0F)+CHR$(&H1F)
30 B$=CHR$(&H07)+CHR$(&H07)+CHR$(&H07)+CHR$(&H07)+
CHR$(&H0F)+CHR$(&H1F)+CHR$(&H19)+CHR$(&H10)
40 C$=CHR$(&H00)+CHR$(&H80)+CHR$(&HCO)+CHR$(&HCO)+
CHR$(&HCO)+CHR$(&HCO)+CHR$(&HE0)+CHR$(&HFO)
50 D$=CHR$(&HCO)+CHR$(&HCO)+CHR$(&HCO)+CHR$(&HCO)+
CHR$(&HE0)+CHR$(&HFO)+CHR$(&H30)+CHR$(&H10)
60 SPRITE$(1)=A$+B$+C$+D$

```

```
10 SCREEN 2,0
20 DATA 0,0,0,108,146,16,0,0
30 A$=""
40 FOR I=1 TO 8
50 READ A:A$=A$+CHR$(A)
60 NEXT I
70 SPRITE$(1)=A$
```

59

O nosso grande "sprite" pode ser codificado como segue:

```
10 SCREEN 2,2
20 DATA 1,3,7,7,7,7,15,31
30 DATA 7,7,7,7,15,31,25,16
40 DATA 0,128,192,192,192,192,224,240
50 DATA 192,192,192,192,224,240,48,16
60 A$=""
70 FOR I=1 TO 32
80 READ A:A$=A$+CHR$(A)
90 NEXT I
100 SPRITE$(1)=A$
```

Neste exemplo o valor decimal de todos os 32 bytes é armazenado em 4 instruções "DATA". Estes valores são lidos na linha 80 e armazenados na variável "A\$" como valores alfanuméricos.

Como colocar "sprites" no écran

Um código é colocado na instrução "SCREEN" imediatamente a seguir ao modo de écran, que indica que "sprites" irão ser usados. O código usado tem o seguinte significado:

- 0 = pequenos "sprites", formato 8×8
- 1 = pequenos "sprites", alargados para formato 16×16
- 2 = grandes "sprites", formato 16×16
- 3 = grandes "sprites", alargados para formato 32×32

Na linha seguinte do programa indicamos primeiro o modo de écran (2) seguido da informação de que serão usados "sprites" de 32×32 .

```
10 SCREEN 2,3
```

O que ainda não referimos até aqui, mas que se torna evidente pelo que foi dito acima é que os "sprites" podem ser aumentados para o dobro do tamanho original.

Há algumas restrições respeitantes ao uso de "sprites":

- a. Os "sprites" não podem ser usados nos modos de texto 1 e 2
- b. Não se podem misturar "sprites" de tamanhos diferentes
- c. Um máximo de 4 "sprites" pode ser simultaneamente colocado numa linha horizontal no écran

Um "sprite" é colocado no écran através da instrução "PUT SPRITE", seguida da seguinte informação:

- a. Número de prioridade
- b. Posição no écran
- c. Cor do "sprite"
- d. Número do "sprite"

O número de prioridade deve ser um número de 0 a 31 e tem o seguinte significado: Quando dois "sprites" são colocados na mesma posição no écran, o "sprite" com o número de prioridade mais baixo terá prioridade sobre o sprite com o número de prioridade mais alto.

A posição no écran é indicada pelas coordenadas X e Y. A coordenada X pode ser um número de - 32 a 255 e a coordenada Y um número de - 32 a 191. É portanto possível colocar um "sprite" fora do écran.

O X e Y indicam a imagem superior e mais à esquerda do "sprite".
As coordenadas podem ter um valor relativo ou absoluto. Quando a coordenada Y tem um valor de 209, o sprite desaparecerá do écran.

A cor do "sprite" é indicada pela cor de código (ver Apêndice C).

O número do "sprite" determina finalmente qual é o "sprite" que deverá ser colocado no écran.

Páre-se o programa em curso (premindo as teclas CTRL + STOP) e escreva-se o seguinte exemplo:

```
NEW
10 SCREEN 2,0
20 DATA 0,3,31,63,255,255,63,7
30 DATA 0,128,248,255,255,254,252,192
40 DATA 0,0,0,108,146,16,0,0
50 FOR I=1 TO 3
60 A$=""
70 FOR J=1 TO 8
80 READ A:A$=A$+CHR$(A)
90 NEXT J
100 SPRITE$(I)=A$
110 NEXT I
120 PUT SPRITE 4,(180,50),15,1
130 PUT SPRITE 5,(188,50),15,2
140 FOR I=-32 TO 212
150 PUT SPRITE 3,(I,50),1,3
160 NEXT I
170 PUT SPRITE 3,(I,209)
180 FOR I=212 TO -32 STEP -1
190 PUT SPRITE 3,(I,50),1,3
200 NEXT I
210 GOTO 140
RUN
```

Se tudo foi correctamente feito, ver-se-á um pássaro preto voando sobre uma nuvem branca. A nuvem branca foi definida por dois "sprites" ("sprites" n.ºs 1 e 2) enquanto que o pássaro preto é definido com o "sprite" n.º 3).

A nuvem branca é projectada no écran pelas instruções nas linhas 120 e 130 do programa. O pássaro negro move-se da esquerda para a direita dirigido por "FOR...NEXT" nas linhas 140 a 160. O "FOR...NEXT" nas linhas 180 a 200 faz com que o pássaro voe novamente da direita para a esquerda.

Páre-se o programa (premindo as teclas CTRL + STOP) e mude-se a linha 10 para:

```
10 SCREEN 2,1
```

Faça-se correr o programa novamente. O tamanho dos "sprites" duplicou agora, como se pode ver.

Páre-se o programa novamente (CTRL + STOP) e mude-se a linha 190 para:

```
190 PUT SPRITE 6,(I,50),1,3
```


Corra o programa mais uma vez. O pássaro continua a voar da esquerda para a direita, mas desta vez desaparece atrás da nuvem, uma vez que agora foi dado ao pássaro um número de prioridade mais baixo que à nuvem na linha 130.

Colisão de "sprites"

O MSX-BASIC torna possível a execução de uma dada sub-rotina quando dois "sprites" colidem. Isto é feito com a instrução "ON SPRITE GOSUB".

Páre-se o programa em curso (premindo a teclas CTRL+STOP) e mude-se o último programa da seguinte maneira:

```
5 SCREEN 2,0
10 ON SPRITE GOSUB 300
135 SPRITE ON
210 GOTO 135
300 PUT SPRITE 3,(I,209)
310 FOR J=50 TO 191
320 PUT SPRITE 3,(I,J),1,3
325 NEXT J
330 I=0:SPRITE OFF
340 RETURN 140
```

Corram de novo o programa. A linha 10 indica agora a sub-rotina para a qual o MSX-BASIC tem que saltar quando dois "SPRITES" se tocam. Na linha 135 do programa a função "collide" (colisão) é activada. Na sub-rotina das linhas 300 a 340 o pássaro é dirigido para baixo por "FOR...NEXT". Na linha 330 a função colisão é indicada e a linha 340 dá instruções ao computador para saltar para trás, da sub-rotina para a linha 140 do programa principal.

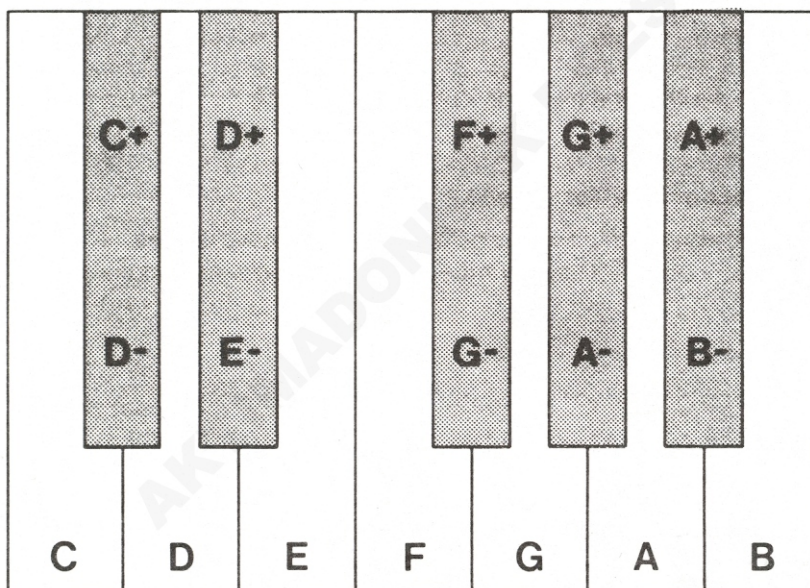
Pode parar-se o programa premindo simultaneamente as teclas CTRL e STOP como já fizemos muitas vezes.

Música

Capítulo 2.13 MSX-BASIC tem uma instrução "PLAY", quando se pretende usar o computador para fazer música. Segundo esta instrução podem introduzir-se os tons para três vozes. A música para cada uma destas três vozes entra através de um número de subcomandos, tais como a instrução "DRAW".

1. Subcomando para tocar as notas de uma oitava

As letras A a G são usadas para indicar que notas devem ser tocadas. Um sinal (+) aumenta meio tom uma nota ($A + = A$ agudo), enquanto um menos (-) produz um tom que é meia nota mais grave ($A \times = A$ bemol).



Basicamente resulta como no teclado de um piano, o que significa que "+" só pode ser usado quando a tecla correspondente do piano é seguida duma tecla preta e um "-" pode ser usado apenas quando é precedido de uma tecla preta. Combinações como "F-" ou "E+" não são permitidas.

Exemplo:

PLAY "CEG"

2. Subcomando para colocar uma oitava

O tom duma nota é decidido pela subcomando "On" sendo "n" qualquer número de 1 a 8. Uma oitava standard (como no piano) vai de "C" a "B".

Exemplo:

PLAY "05CEG04CEG05CEG"

3. Subcomando para tocar notas

Em vez de fazer entrar uma oitava mais uma nota, é também possível fazer entrar um número de nota com o subcomando "Nn", sendo "n" qualquer número entre 0 e 96. Os dois exemplos seguintes têm exactamente o mesmo efeito:

PLAY "03CEG"
PLAY "N24N28N31"

4. Subcomando para extensão da nota

O subcomando "Ln" é usado para determinar a extensão duma nota. Nesta combinação, "n" é um número entre 1 e 64.

O número tem o seguinte significado:

- 1 = nota completa
- 2 = metade da nota
- 3 = um terço da nota
- 4 = um quarto da nota
- etc.

Exemplo:

PLAY "L8DEL4DCE"

5. Subcomando para pausas

Uma pausa é introduzida quando o subcomando "Rn" é introduzido. Aqui, de novo, "n" é um número entre 1 e 64. Estes números representam as mesmas extensões, como no caso do subcomando "Ln".

6. Subcomando para mudar a extensão duma nota

Colocando um ponto final (.) atrás da nota ela é prolongada por $\frac{1}{2}$ da sua extensão original. É também possível usar mais que um ponto à frente da nota. Exemplo:

PLAY "A..."

Neste exemplo a nota "A" soará $\frac{15}{8}$ vezes a sua extensão normal ($= 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8}$).

7. Subcomando para o tempo

O tempo é determinado pelo subcomando "Tn", sendo "n" um número entre 32 e 255. Representa o número de quartos de nota durante um minuto.

8. Subcomando para o volume

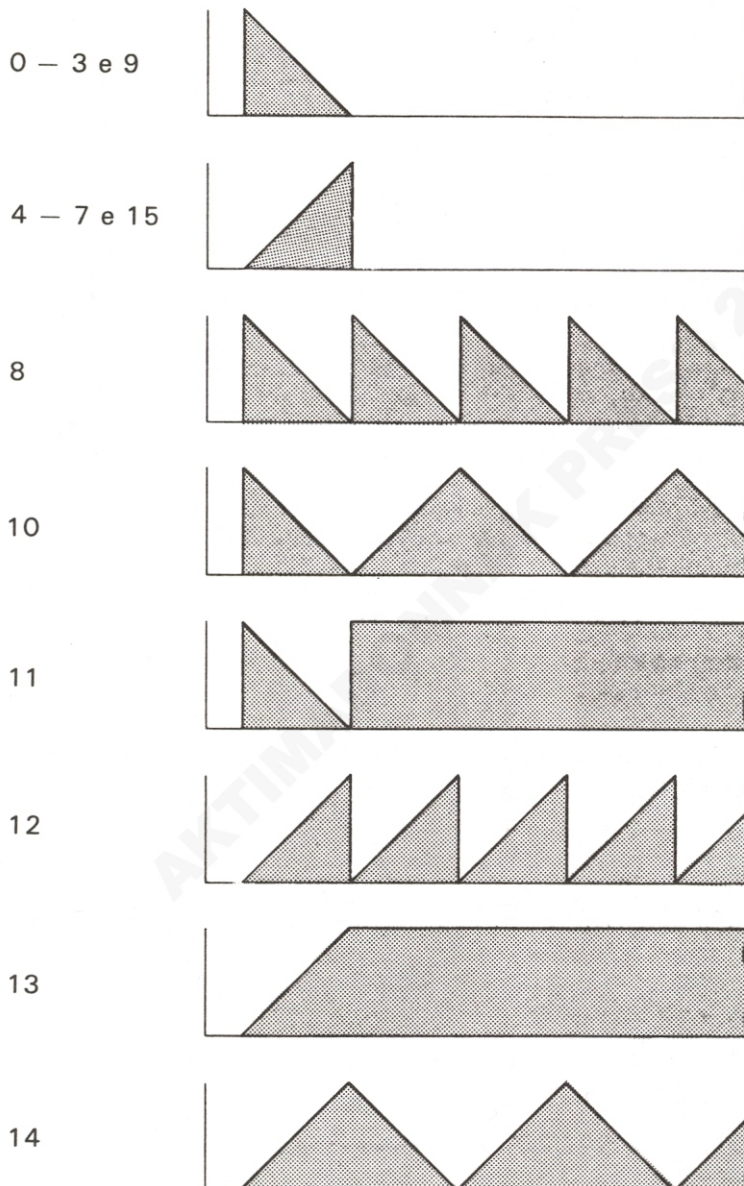
O volume dum tom é decidido pelo subcomando "Vn" representando "n" um número entre 0 e 15, com o último representando o volume mais alto.

Exemplo:

PLAY "V15CGR1V9CG"

9. Subcomando para a forma do som

O subcomando "Sn" é usado para escolher a forma do tom. "n" pode ser um número entre 0 e 15:



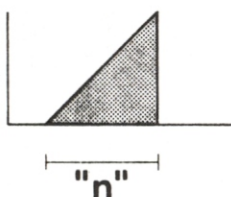
10. Subcomando para a extensão do tom agudo

"Mn" é o subcomando usado para escolher a extensão do tom agudo.

O "n" representa qualquer número entre 1 e 65535. Oíça apenas a diferença entre os seguintes exemplos:

```
PLAY "S15M500C"  
PLAY "S15M1000C"  
PLAY "S12M500C"  
PLAY "S12M1000C"
```

Quanto mais baixo for o número depois do "M", mais curta é a forma.



11. Como executar os subcomandos

O MSX-BASIC torna possível colocar todos os subcomandos numa variável alfanumérica e referir-se a esta variável na instrução "PLAY". O último exemplo das séries acima representadas podem também ser escritas da seguinte maneira:

```
NEW  
10 A$=S12M1000C"  
20 PLAY "XA$;"  
RUN
```

Um ponto e vírgula deve ser colocado imediatamente depois do nome da variável.

Todas as constantes usadas no subcomando da instrução "PLAY" podem ser substituídas da seguinte maneira: "= nome da variável".

Aqui temos um exemplo: X1=15

```
PLAY "N=X1;"
```

Todos os subcomandos podem ser usados para três vezes com a instrução "PLAY". Uma vírgula (,) deve ser colocada entre os subcomandos de cada voz. Por exemplo "PLAY", "O4A", "O5C", "O3B".

A instrução "PLAY" oferece muitas possibilidades. Experimentem todas.

Aqui temos outro exemplo:

```
NEW  
5 CLEAR 1000  
10 A$="V8S1M9000T250L405"  
20 B$="F+F+8E2R8EE8D2R8R4F+F+F+G2F+"  
30 C$="04B05DD04B05D2E04B05D"  
40 D$="04B.05DR4.B.A.DD1R2."  
50 H$=B$+".R8"+B$+"04A"+C$+"DDD8C+2R404A"+C$+  
".D8D.D8E.R8R2"+B$+".R8"+D$  
60 I$="R1R1R4AAAB2A.R8R1R1R4AAAB2AR1R1R1R1R1R1R1R1R1R1R4R8AAAB2A"  
70 PLAY "XA$;","XA$;"
```

```
80 PLAY "XH$;" , "XI$;"  
RUN
```

Talvez tenha reconhecido o tom.

Com a função "PLAY" (não haja confusão com a instrução "PLAY") verificar se a instrução "PLAY" executada, pode tentar-se juntando as seguintes linhas ao programa anterior:

```
85 CLS  
90 PRINT PLAY(1):GOTO 90
```

Corra o programa novamente. Enquanto a melodia está a ser tocada, o valor - 1 é impresso no écran. Pode parar-se o programa premindo simultaneamente as teclas CTRL e STOP.

Com a instrução "SOUND" um valor específico pode também ser colocado num dos registos do "chip" de som. Só deve usar esta instrução quando já estiver familiarizado com o modo como trabalhar o "chip" do som.

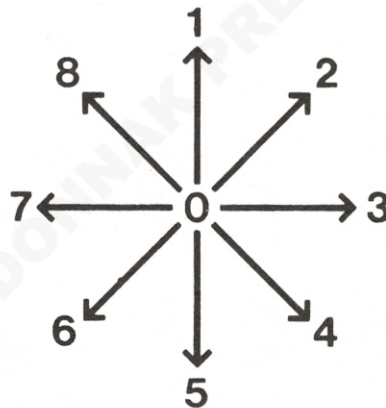
MSX-BASIC e os comandos manuais

Capítulo 2.14 Pode ligar-se um ou dois comandos manuais ao computador MSX. O MSX-BASIC pode responder na direcção em que a alavanca se desloca. Podem também usar-se as teclas de controlo do cursor para este fim. A direcção da alavanca pode ser verificada com a função "STICK(n)". "n" deve ser um número de 0 a 2 e tem os seguintes significados:

- 0 = As teclas de controlo do cursor são usadas como comandos
- 1 = Comando manual ligado à tomada 1
- 2 = Comando manual ligado à tomada 2

Os valores obtidos com a função "STICK(n)" vão de 0 a 8:

- 0 = Neutro
- 1 = Norte
- 2 = Noroeste
- 3 = Este
- 4 = Sudeste
- 5 = Sul
- 6 = Sudoeste
- 7 = Oeste
- 8 = Noroeste



Aqui está um exemplo:

```
NEW
10 A=STICK(0)
20 IF A=STICK(0) THEN 20
30 PRINT STICK(0)
40 GOTO 10
RUN
```

Neste exemplo a variável "A" na linha 10 do programa é igual no valor das teclas de controlo do cursor. Enquanto nenhuma das teclas de controlo do cursor for premida, o valor de "A" será portanto de zero. Quando for premida uma das teclas de controlo do cursor, a linha 30 será executada. Experimentemos todas as teclas de controlo do cursor.

Pode parar-se o programa com as teclas CTRL e STOP.

O MSX-BASIC pode também entrar em funcionamento se o botão de acção do comando manual for premido. Para este fim pode também utilizar-se a barra de

espaços do teclado. Isto pode ser feito através da função "STRIG(n)" sendo o "n" um número de 0 a 4. Estes valores têm o seguinte conteúdo:

0 = A barra de espaços é o botão de acção

1 = Botão de acção 1 do comando manual ligado à entrada 1

2 = Botão de acção 1 do comando manual ligado à entrada 2

3 = Botão de acção 2 do comando manual ligado à entrada 1

4 = Botão de acção 2 do comando manual ligado à entrada 2

O valor obtido com a função "STRIG(n)" é 0 quando o botão de acção não é premido e -1 quando o botão for premido.

Exemplo:

```
NEW
10 IF STRIG(0)=-1 THEN PRINT STRIG(0)
20 GOTO 10
RUN
```

Na linha 10 o MXS-BASIC verifica se a barra de espaços foi premida.

Se tiver sido, o valor de "STRIG(0)" é projectado no écran.

Pare-se o programa com as teclas CTRL e STOP.

Lançamento de um foguete

No exemplo a seguir um foguete é lançado premindo a barra de espaços. Escreva-se atentamente o programa que segue para não cometer erros!

```
NEW
10 SCREEN 2,3,:COLOR 1,4,1
20 DATA 0,0,0,0,3,7,15,63
30 DATA 63,127,127,63,31,1,0,0
40 DATA 0,0,0,0,192,252,255,255
50 DATA 255,255,255,255,255,255,28,0
60 DATA 0,0,0,1,15,31,255,255
70 DATA 255,255,255,255,255,255,30,0
80 DATA 0,0,0,128,224,248,248,252
90 DATA 252,248,248,240,240,224,0,0
100 DATA 0,0,0,28,126,127,255,127
110 DATA 31,15,7,3,0,0,0,0
120 DATA 0,0,0,0,96,240,248,252
130 DATA 254,254,252,252,248,0,0,0
140 DATA 0,0,0,0,0,0,3,15
150 DATA 63,255,255,127,63,31,0,0
160 DATA 0,0,0,0,0,248,252,254
170 DATA 254,252,252,252,252,248,248,0
180 DATA 0,0,0,0,0,0,0,0
190 DATA 0,0,0,0,0,0,0,0
200 DATA 3,3,7,7,7,7,7,15
210 DATA 15,15,15,31,31,31,31,31
220 DATA 0,0,0,0,0,0,0,0
230 DATA 0,0,0,0,0,0,0,0
240 DATA 255,255,255,16,40,16,0,0
250 DATA 0,0,0,0,0,0,0,0
260 DATA 255,255,255,0,1,0,0,0
270 DATA 0,0,0,0,0,0,0,0
280 DATA 240,240,240,128,64,128,0,0
290 DATA 0,0,0,0,0,0,0,0
```



```

300 DATA 8,28,28,62,62,62,127,255
310 DATA 62,62,62,62,127,255,201,128
320 DATA 0,0,0,0,0,0,0,128
330 DATA 0,0,0,0,0,128,128,128
340 DATA 8,28,28,62,62,62,62,28
350 DATA 28,20,0,0,0,0,0,0
360 DATA 0,0,0,0,0,0,0,0
370 DATA 0,0,0,0,0,0,0,0
380 FOR J=1 TO 9:A$=""
390 FOR I=1 TO 32:READ A:A$=A$+CHR$(A):NEXT I
400 SPRITE$(J)=A$:NEXT J
410 CIRCLE (88,120),16,6,-0.0001,-3.415,0.5
420 PAINT (88,118),6
430 LINE (0,192)-(255,120),12,BF
440 PUT SPRITE 1,(80,16),15,1
450 PUT SPRITE 2,(112,16),15,2
460 PUT SPRITE 10,(160,80),15,3
470 PUT SPRITE 11,(16,48),15,4
480 I=120:J=144:GOSUB 1000
490 IF STRIG(0)=-1 GOTO 2000
500 IF STICK(0)=3 THEN GOSUB 3000
510 IF STICK(0)=7 THEN GOSUB 4000
520 GOTO 490
1000 PUT SPRITE 5,(I,J),1,5:J=J+32
1010 PUT SPRITE 6,(I,J),1,6:I=I+32
1020 PUT SPRITE 7,(I,J),1,7:J=J-32
1030 PUT SPRITE 8,(I,J),7,8:I=I-32
1040 RETURN
2000 I=I+32:FOR K=144 TO -33 STEP -1
2010 PUT SPRITE 8,(I,K),7,8:M=K+32
2020 PUT SPRITE 9,(I,M),9,9
2025 FOR T=1 TO 1:NEXT T
2030 PUT SPRITE 9,(I,209)
2040 NEXT K:FOR K=1 TO 1000:NEXT K:COLOR 15,4,7:END
3000 I=I+1:IF I>200 THEN I=200
3010 GOSUB 1000:RETURN
4000 I=I-1:IF I<0 THEN I=0
4010 GOSUB 1000:RETURN

```

Corra o programa. Pode deslocar-se a plataforma de lançamento para a esquerda ou para a direita com as teclas de controle do cursor. O foguete é lançado quando for premida a barra de espaços. Se o programa não funcionar devidamente, verifique-se cuidadosamente se todas as instruções foram devidamente escritas.

Capítulo 3

MSX HARDWARE

3.1 Manutenção

3.2 Periféricos

AKTIMADONNAK PRESS 2013

Manutenção

Capítulo 3.1 O VG8010 pode ser limpo com um pano seco. Nunca se devem utilizar produtos químicos!

Se se usar um gravador de dados, as cabeças do gravador devem ser limpas regularmente, seguindo as instruções que o acompanham.

Asseguremo-nos de que as cassettes são guardadas num local fresco, mantendo-as afastadas da luz solar ou de outras fontes de calor, bem como de quaisquer campos magnéticos. Se não o fizermos arriscamo-nos a perder todas as informações armazenadas nas cassettes.

Se o VG8010 não funcionar devidamente (se se ouvir um ruído estranho ou cheiro devido a sobreaquecimento) desligue-se imediatamente o computador e leve-se o mesmo ao vendedor para uma verificação completa.

Todas as reparações devem ser feitas por ou através do vendedor. Nunca se deve abrir uma consola.

O que não se deve fazer

Como qualquer outro aparelho eléctrico e electrónico, o VG8010 é alérgico ao excesso de humidade! Nunca se deve deixar cair qualquer espécie de líquido em cima do computador!

O VG8010 tem as necessárias aberturas para a ventilação e arrefecimento. Asseguremo-nos de que as aberturas ficam livres com vista a manter a circulação de ar indispensável.

Mantenha-se o VG8010 afastado de todas as fontes de calor como fogões, irradiadores e até mesmo luz do sol.

Nunca se toquem nos pontos de contacto das ligações com os dedos, já que isso pode provocar corrosão desnecessária.

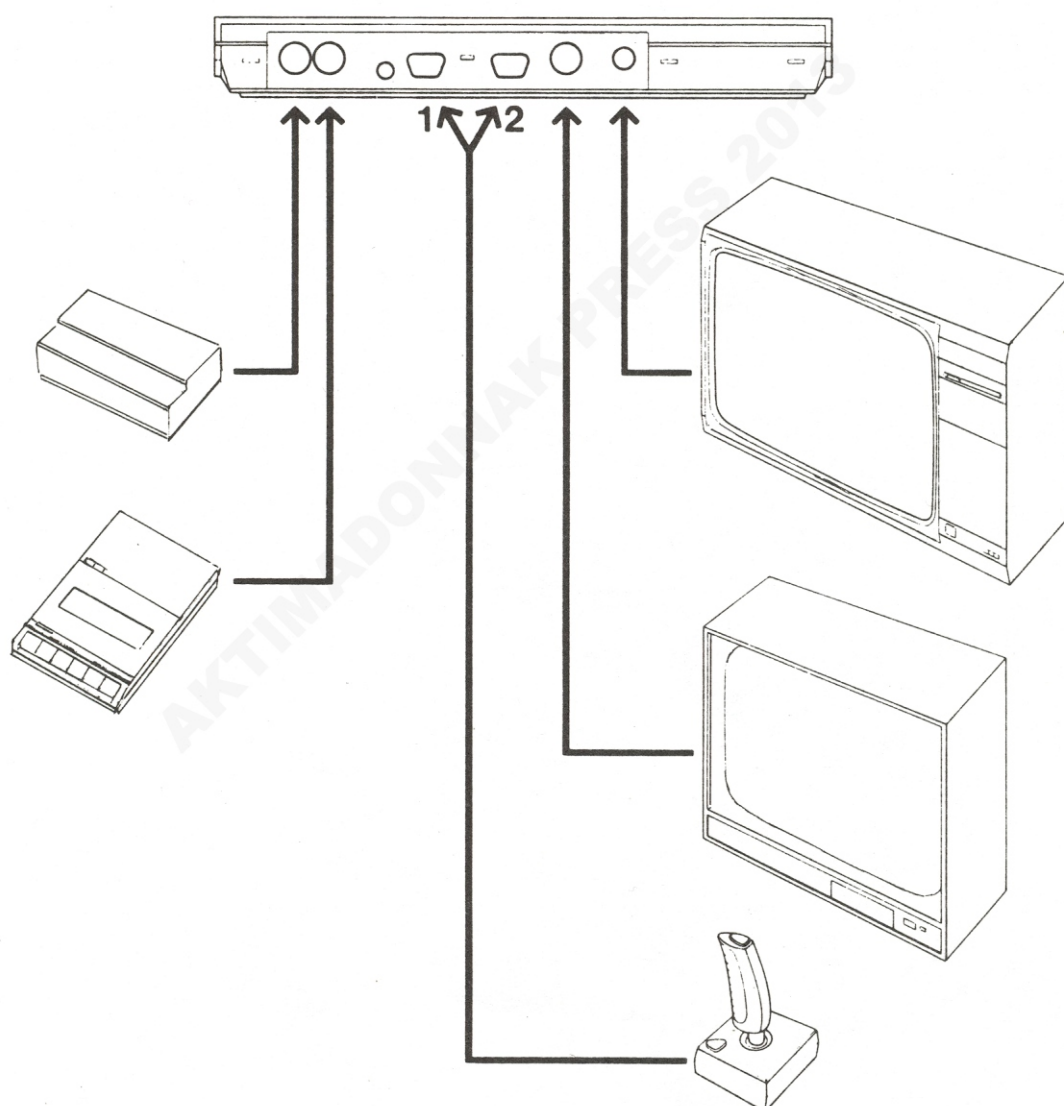
Coloquem-se os cabos de ligação de tal maneira que ninguém tropece neles.

Ao retirar ou desligar uma ficha, não puxe o cabo mas sim a ficha que deve ter bem segura na mão.

Não deixe cair o VG8010 ao deslocá-lo e nunca coloque nada pesado em cima da consola.

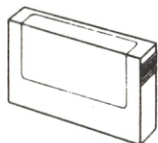
Periféricos

Capítulo 3.2 Na parte de trás do VG8010 encontram-se as ligações standard para o seguinte equipamento periférico e acessórios:

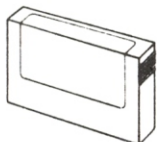


Nota: Pode utilizar-se outro equipamento (ver instruções fornecidas com os respectivos periféricos).

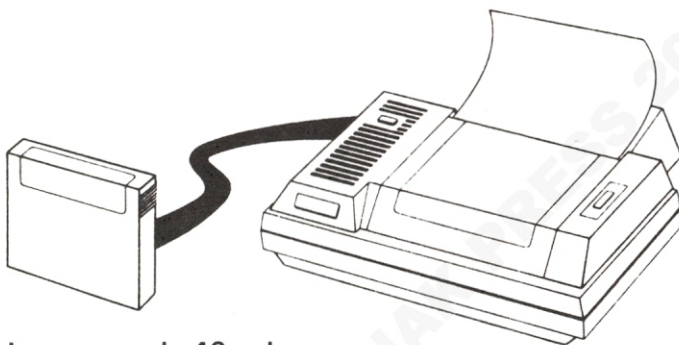
Entre outros, os periféricos seguintes podem ser ligados através das aberturas de inserção das "cartridges":



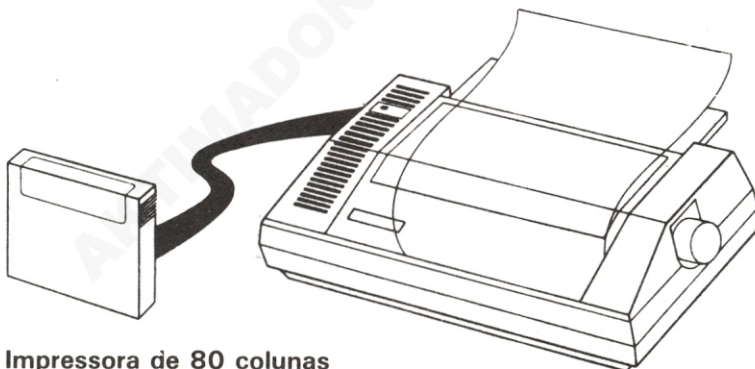
Cartridge de 16K RAM



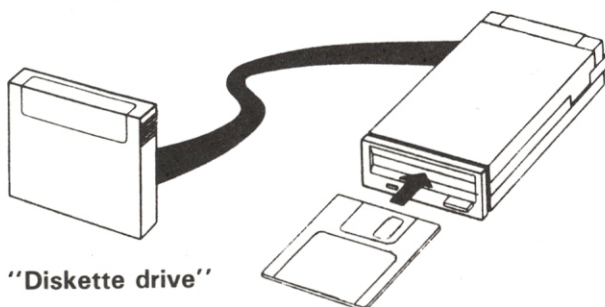
Cartridge ROM de programa



Impressora de 40 colunas



Impressora de 80 colunas



"Diskette drive"

Capítulo 4

Apêndices

- A Lista de mensagens com erro
- B Funções matemáticas
- C Tabela de cores
- D Funções de controle
- E Códigos de caracteres
- F Palavras reservadas
- G Teclado
- H Especificações técnicas
- I Processador Video
- J Gerador de som programável

Apêndice A

Lista de erros de mensagem

Erros	Código	Explicação
Nome de ficheiro errado	56	Foi dado um nome incorrecto para indicar um ficheiro.
Número de ficheiro errado	52	O número de ficheiro usado refere-se a um ficheiro que ainda não foi aberto com a instrução "OPEN", ou o número de ficheiro é mais alto que o mais alto número introduzido com a instrução "MAXFILES".
Não pode continuar	17	O operador está a tentar continuar com um programa que: — foi interrompido por um erro de mensagem — foi alterado enquanto a sua execução foi interrompida — não existe.
"Device" 1/O erro	19	Um erro de input ou output descoberto ao ler ou escrever um ficheiro ou programa.
Declaração directa (Direct Statement)	57	Uma declaração directa é encontrada durante o processo de entrada dum ficheiro ASCII.
Divisão por zero	11	Enquanto procede a um cálculo o computador encontra uma divisão por 0 ou uma situação em que 0 é levantado a uma potência negativa.
"Field overflow"	50	O número total de bytes colocados na instrução "FIELD" é maior que 256.
Arquivo já aberto	54	É pedido ao computador que abra um arquivo com a instrução "OPEN" quando esse ficheiro já foi aberto. Ou uma instrução "KILL" é dada para um arquivo que já está aberto.
Ficheiro não aberto	59	O operador está a referir-se a um ficheiro com as instruções ler ou escrever, o qual ainda não foi aberto com a instrução "OPEN".

"Illegal Direct"	12	Uma declaração ilegal é usada no modo directo.
Função de chamada ilegal	5	Um parâmetro errado é dado a uma função. Isto pode acontecer como resultado de: — uma negativa ou um texto demasiado grande — uma negativa ou um parâmetro zero com a função "LOG". — um parâmetro negativo com a função "SQR" — um parâmetro incorrecto com "MID\$", "LEFT\$", "RIGHT", "INP", "OUT", "PEEK", "POKE", "TAB", "SPC", "STRING\$", "SPACE\$", "INSTR\$", "ON... GOTO", "ON... GOSUB".
"INPUT" passou do fim	55	O operador está a tentar ler dados de um ficheiro que já foi completamente lido. Use a função "EOF" para evitar este erro.
Erro interno	51	Uma situação inesperada ocorreu, a qual não pode ser resolvida pelo interpretador BASIC.
"Line buffer overflow"	25	O operador escreveu uma linha contendo demasiados caracteres.
"Missed operand"	24	Uma expressão contém uma função, instrução ou comando sem um parâmetro.
"NEXT sem FOR"	1	Uma variável numa instrução "NEXT" não corresponde à variável da instrução "FOR" precedente ou uma instrução "NEXT" não é precedida por uma "FOR".
"no RESUME"	21	Um erro não está terminado por uma instrução "RESUME".
"Out of DATA"	4	É executada uma instrução "READ" quando já não há instrução "DATA" que não tenha já sido lida.
"Out of Memory"	7	Um programa é demasiado grande ou contém excesso de "loops" "FOR", "GOSUB" ou variáveis. Pode também conter expressões demasiado complicadas.
"Out of string space"	14	As variáveis alfanuméricas excedem o espaço disponível. Utilize a instrução "CLEAR" para criar espaço adicional.
"Overflow"	6	O resultado de um cálculo é demasiado grande para ser reproduzido.
"Redimensional array"	10	Duas instruções "DIM" foram dadas à mesma variável já em uso e dimensionada com a instrução "DIM".

"RESUME sem erro"	22	É pedido ao computador para executar uma instrução "RESUME" apesar de não ter sido executada nenhuma instrução "ON ERROR GOTO".
"RETURN sem GOSUB"	3	Uma instrução "RETURN" é encontrada para a qual não há declaração prévia "GOSUB".
"Sequential 1/0 only"	58	O operador está a tentar ler ou escrever ao acaso num ou a partir de um arquivo sequencial.
Fórmula em cadeia demasiado complexa	16	A expressão alfanumérica é demasiado longa ou demasiado complexa.
Cadeia demasiado longa	15	Foi feita uma tentativa para pôr mais que 255 caracteres alfanuméricos numa cadeia.
"Subscript out of range" (Endereço fora de alcance)	9	Fez-se uma tentativa para se referir a uma cadeia de expressões com um endereço errado.
"Syntax error" (Erro de sintaxe)	2	Uma declaração (ordem, instrução ou função) contém caracteres incorrectamente colocados, pontuação mal feita, etc.
"Type mismatch"	13	É dada a uma variável alfanumérica um valor numérico ou vice-versa, ou a uma função que deveria ter um valor numérico foi dado um parâmetro alfanumérico.
Número de linha não identificável	13	Uma ordem ou instrução foi dada referindo-se a uma linha não existente.
Função do utilizador não identificável	18	Uma função do operador foi chamada antes de ser definida a função com a instrução "DEF"
Erro não impresso	23-26 49-60 255	Estes códigos ainda não foram definidos. O operador pode utilizá-los para colocar as suas próprias definições de erro.
Erro a verificar	20	Este erro é dado quando acontece que o programa na memória difere do da cassette depois de executada a instrução "CLOAD?"

Apêndice B

Funções matemáticas

Funções matemáticas que não foram incluídas no MSX-BASIC são definidas do modo seguinte:

Derived functions	MSX-BASIC definition
SECANT	$= 1/\text{COS}(X)$
COSECANT	$= 1/\text{SIN}(X)$
COTANGENT	$= 1/\text{TAN}(X)$
INVERSED SINE	$= \text{ATN}(X/\text{SQR}(-X \cdot X + 1))$
INVERSED COSINE	$= -\text{ATN}(X/\text{SQR}(-X \cdot X + 1)) + 1.5708$
INVERSED SECANT	$= \text{ATN}(X/\text{SQR}(X \cdot X - 1)) + \text{SGN}(\text{SGN}(X) - 1) \cdot 1.5708$
INVERSED COSECANT	$= \text{ATN}(X/\text{SQR}(X \cdot X - 1)) + (\text{SGN}(X) - 1) \cdot 1.5708$
INVERSED COTANGENT	$= \text{ATN}(X) + 1.5708$
HYPERBOLIC SINE	$= (\text{EXP}(X) - \text{EXP}(-X))/2$
HYPERBOLIC COSINE	$= (\text{EXP}(X) + \text{EXP}(-X))/2$
HYPERBOLIC TANGENT	$= \text{EXP}(-X)/\text{EXP}(X) + \text{EXP}(-X) \cdot 2 + 1$
HYPERBOLIC SECANT	$= 2/(\text{EXP}(X) + \text{EXP}(-X))$
HYPERBOLIC COSECANT	$= 2/(\text{EXP}(X) - \text{EXP}(-X))$
HYPERBOLIC COTANGENT	$= \text{EXP}(-X)/(\text{EXP}(X) - \text{EXP}(-X)) \cdot 2 + 1$
INV. HYPERBOLIC SINE	$= \text{LOG}(X + \text{SQR}(X \cdot X + 1))$
INV. HYPERBOLIC COSINE	$= \text{LOG}(X + \text{SQR}(X \cdot X - 1))$
INV. HYPERBOLIC TANGENT	$= \text{LOG}((1 + X)/(1 - X))/2$
INV. HYPERBOLIC SECANT	$= \text{LOG}((\text{SQR}(-X \cdot X + 1) + 1)/X)$
INV. HYPERBOLIC COSECANT	$= \text{LOG}((\text{SGN}(X) \cdot \text{SQR}(X \cdot X + 1) + 1)/X)$
INV. HYPERBOLIC COTANGENT	$= \text{LOG}((X + 1)/(X - 1))/2$

Apêndice C

Tabela de cores

Número da Cor	Cor
0	Transparente
1	Preto
2	Verde
3	Verde claro
4	Azul escuro
5	Azul claro
6	Vermelho escuro
7	Azul ciânico
8	Vermelho
9	Vermelho claro
10	Amarelo torrado
11	Amarelo claro
12	Verde escuro
13	Magenta
14	Cinzentos
15	Branco

Apêndice D

Funções de controlo

Na tabela a seguir indicada encontram-se em sumário todas as funções de comando do MSX-BASIC. Estas funções são executadas premindo a tecla "CTRL" e outra, simultaneamente.

Cada função de controlo tem um código. Esse código pode ser usado, por exemplo, na função "CHR\$".

Exemplo:

```
PRINT CHR$(7)
```

Isto produzirá um "beep" (som)

No fim desta tabela encontra-se uma descrição das várias funções de controlo:

CÓDIGO	Tecla CTRL + TECLA:	TECLA ESPECIAL	FUNÇÃO
1	A		Caracter alternativo
2	B		Cursor para a palavra anterior
3	C		Parar o comando AUTO
4	D		Nenhuma
5	E		Linhas truncadas
6	F		Cursor para a palavra seguinte
7	G		Som (beep)
8	H	BS	Espaço para trás
9	I	TAB	Tabular
10	J		Linha seguinte
11	K	HOME	Cursor para canto superior esquerdo
12	L	CLR	Limpar écran
13	M	RETURN	Mudar de linha
14	N		Cursor para o fim da linha
15	O		Nenhuma
16	P		Nenhuma
17	Q		Nenhuma
18	R	INS	Inserir

19	S		Nenhuma
20	T		Nenhuma
21	U		Limpar a linha
22	V		Nenhuma
23	W		Nenhuma
24	X	SELECT	Nenhuma
25	Y		Nenhuma
26	Z		Nenhuma
27	[	ESC	Nenhuma
28	\	→	Cursor para a direita
29	]	←	Cursor para a esquerda
30	^	↑	Cursor para cima
31	_	↓	Cursor para baixo

Descrição das funções de controlo

Caracteres alternativos	Com esta função o carácter seguinte será um carácter de alternativa (ver apêndice E)
Cursor para a palavra precedente	O cursor deslocar-se-á para o primeiro carácter da palavra anterior. Para o MSX-BASIC o primeiro carácter é de A a Z, a a z ou 0 a 9.
Comando Stop Auto	Esta função traz o MSX-BASIC ao nível de comando se o comando "AUTO" tiver sido activado.
Interrupção da linha	Todos os caracteres e símbolos serão removidos, começando do cursor até ao fim da linha.
Cursor para a palavra seguinte	O cursor desloca-se para o primeiro carácter da palavra seguinte. Para o MSX-BASIC o primeiro carácter é A a Z, a a z ou 0 a 9.
Som "beep"	Esta função produzirá um som "beep".
Espaço para trás	Esta função de controlo deslocará um carácter para imediatamente à esquerda do cursor. O próprio cursor e todos os caracteres a seguir ao cursor deslocar-se-ão uma posição para a esquerda.
Tabular	O cursor desloca-se para a próxima paragem do tabulador. Há uma paragem do tabulador em cada 8 posições.
Linha seguinte	O cursor desloca-se para a primeira posição da linha seguinte.

Cursor para o canto superior esquerdo	O cursor é deslocado para o canto superior esquerdo do écran sem limpar o écran.
Limpar o écran	O cursor é deslocado para o canto superior esquerdo e o écran fica limpo.
“Carriage return” (Retorno automático)	As expressões (ou frases) são automaticamente transferidas no MSX BASIC.
Cursor para o fim da linha	O cursor é deslocado para a posição a seguir ao último carácter da linha.
Inserir	Inserem-se caracteres adicionais entre caracteres já existentes. Se esta função de controlo for repetida, os novos caracteres substituirão os já existentes.
Limpar a linha	Toda a linha é retirada.
Cursor para a direita	O cursor desloca-se uma posição para a direita.
Cursor para a esquerda	O cursor é deslocado uma posição para a esquerda.
Cursor para cima	O cursor é deslocado uma linha para cima.
Cursor para baixo	O cursor é deslocado uma linha para baixo.

Apêndice E

Conjunto de caracteres

Neste apêndice encontram-se todos os caracteres que podem ser usados com o MSX-BASIC e os seus correspondentes códigos. Estes podem ser usados, por exemplo, na função "CHR\$". Os dois exemplos seguintes produzirão exactamente o mesmo efeito:

```
PRINT "A"
```

e

```
PRINT CHR$(65)
```

No modo de texto 1 (40 caracteres para uma linha) as duas colunas da direita de pontos de imagem dos caracteres não são mostradas no écran. Tentem o exemplo seguinte:

```
NEW  
10 SCREEN 0  
20 PRINT CHR$(210)  
RUN
```

Agora mude-se a linha 10 como segue:

```
10 SCREEN 1
```

Percorram o programa e notarão a diferença.

Symbol									
Code	32 &H20	33 &H21	34 &H22	35 &H23	36 &H24	37 &H25	38 &H26	39 &H27	40 &H28
Symbol							" data-bbox="641 133 698 168"/>		
Code	41 &H29	42 &H2A	43 &H2B	44 &H2C	45 &H2D	46 &H2E	47 &H2F	48 &H30	49 &H31
Symbol									
Code	50 &H32	51 &H33	52 &H34	53 &H35	54 &H36	55 &H37	56 &H38	57 &H39	58 &H3A
Symbol									
Code	59 &H3B	60 &H3C	61 &H3D	62 &H3E	63 &H3F	64 &H40	65 &H41	66 &H42	67 &H43
Symbol									
Code	68 &H44	69 &H45	70 &H46	71 &H47	72 &H48	73 &H49	74 &H4A	75 &H4B	76 &H4C
Symbol									
Code	77 &H4D	78 &H4E	79 &H4F	80 &H50	81 &H51	82 &H52	83 &H53	84 &H54	85 &H55
Symbol									
Code	86 &H56	87 &H57	88 &H58	89 &H59	90 &H5A	91 &H5B	92 &H5C	93 &H5D	94 &H5E
Symbol									
Code	95 &H5F	96 &H60	97 &H61	98 &H62	99 &H63	100 &H64	101 &H65	102 &H66	103 &H67
Symbol									
Code	104 &H68	105 &H69	106 &H6A	107 &H6B	108 &H6C	109 &H6D	110 &H6E	111 &H6F	112 &H70
Symbol									
Code	113 &H71	114 &H72	115 &H73	116 &H74	117 &H75	118 &H76	119 &H77	120 &H78	121 &H79
Symbol									
Code	122 &H7A	123 &H7B	124 &H7C	125 &H7D	126 &H7E	127 &H7F	128 &H80	129 &H81	130 &H82
Symbol									
Code	131 &H83	132 &H84	133 &H85	134 &H86	135 &H87	136 &H88	137 &H89	138 &H8A	139 &H8B
Symbol									
Code	140 &H8C	141 &H8D	142 &H8E	143 &H8F	144 &H90	145 &H91	146 &H92	147 &H93	148 &H94

Symbol									
Code	149 &H95	150 &H96	151 &H97	152 &H98	153 &H99	154 &H9A	155 &H9B	156 &H9C	157 &H9D
Symbol									
Code	158 &H9E	159 &H9F	160 &HA0	161 &HA1	162 &HA2	163 &HA3	164 &HA4	165 &HA5	166 &HA6
Symbol									
Code	167 &HA7	168 &HA8	169 &HA9	170 &HAA	171 &HAB	172 &HAC	173 &HAD	174 &HAE	175 &HAF
Symbol									
Code	176 &HB0	177 &HB1	178 &HB2	179 &HB3	180 &HB4	181 &HB5	182 &HB6	183 &HB7	184 &HB8
Symbol									
Code	185 &HB9	186 &HBA	187 &HBB	188 &HBC	189 &HBD	190 &HBE	191 &HBF	192 &HCO	193 &HC1
Symbol									
Code	194 &HC2	195 &HC3	196 &HC4	197 &HC5	198 &HC6	199 &HC7	200 &HC8	201 &HC9	202 &HCA
Symbol									
Code	203 &HCB	204 &HCC	205 &HCD	206 &HCE	207 &HCF	208 &HD0	209 &HD1	210 &HD2	211 &HD3
Symbol									
Code	212 &HD4	213 &HD5	214 &HD6	215 &HD7	216 &HD8	217 &HD9	218 &HDA	219 &HDB	220 &HDC
Symbol									
Code	221 &HDD	222 &HDE	223 &HDF	224 &HE0	225 &HE1	226 &HE2	227 &HE3	228 &HE4	229 &HE5
Symbol									
Code	230 &HE6	231 &HE7	232 &HE8	233 &HE9	234 &HEA	235 &HEB	236 &HEC	237 &HED	238 &HEE
Symbol									
Code	239 &HEF	240 &HF0	241 &HF1	242 &HF2	243 &HF3	244 &HF4	245 &HF5	246 &HF6	247 &HF7
Symbol									
Code	248 &HF8	249 &HF9	250 &HFA	251 &HFB	252 &HFC	253 &HFD	254 &HFE	255 &HFF	

Caracteres de alternativa

Além dos símbolos já indicados há um número de outros símbolos indicados pelos códigos 65 a 95 se forem precedidos por "CHR\$(1)".

Exemplo:

```
NEW
10 SCREEN 1
20 PRINT CHR$(65)
30 PRINT CHR$(1)+CHR$(65)
RUN
```

A linha 20 produzirá o caracter regular "65" (= "A") no écran, enquanto que a linha 30 produzirá o caracter alternativa "65" (= um rosto).

Symbol									
Code	65 &H41	66 &H42	67 &H43	68 &H44	69 &H45	70 &H46	71 &H47	72 &H48	73 &H49
Symbol									
Code	74 &H4A	75 &H4B	76 &H4C	77 &H4D	78 &H4E	79 &H4F	80 &H50	81 &H51	82 &H52
Symbol									
Code	83 &H53	84 &H54	85 &H55	86 &H56	87 &H57	88 &H58	89 &H59	90 &H5A	91 &H5B
Symbol									
Code	92 &H5C	93 &H5D	94 &H5E	95 &H5F					

Matriz de caracteres

Cada linha numa matriz de caracteres é dividida em 2 X 4 quadrados. Os quatro quadrados têm os valores de 8, 4, 2 e 1 respectivamente. Quatro quadrados juntos podem ter um valor de 0 a 15:

8 4 2 1					8 4 2 1				
				= 0					= 8
				= 1					= 9
				= 2					= 10 (A)
				= 3					= 11 (B)
				= 4					= 12 (C)
				= 5					= 13 (D)
				= 6					= 14 (E)
				= 7					= 15 (F)

Os valores de 10 a 15 são representados pelas letras A a F. O valor mínimo por linha da matriz de caracteres é 00 e o valor máximo é FF.

AKTIMADONNAK PRESS 2013

Apêndice F

Palavras reservadas

As palavras que se seguem não podem ser usadas como nomes para variáveis:

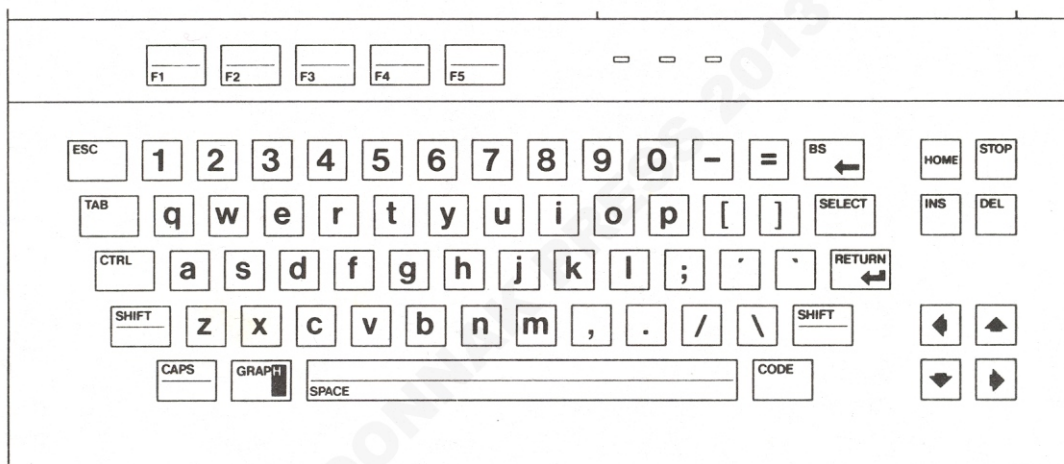
ABS	DATA	IF	NAME	SAVE
AND	DEF	IMP	NEW	SCREEN
AS	DEFINT	INKEY\$	NEXT	SGN
ASC	DEFDBL	INP	NOT	SIN
ATN	DEFSNG	INPUT	OCT\$	SOUND
AUTO	DEFSTR	INPUT\$	OFF	SPACE\$
BASE	DEFUSR	INSTR	ON	SPC
BEEP	DELETE	INT	OPEN	SPRITE
BIN\$	DIM	INTERVAL	OR	SPRITE\$
BLOAD	DRAW	KEY	OUT	SQR
BSAVE	DSKF	KILL	PAD	STEP
CALL	ELSE	LEFT\$	PAINT	STICK
CDBL	END	LEN	PDL	STOP
CHR\$	EOF	LET	PEEK	STR\$
CINT	EQV	LINE	PLAY	STRIG
CIRCLE	ERASE	LIST	POINT	STRING\$
CLEAR	ERL	LLIST	POKE	SWAP
CLOAD	ERR	LOAD	POS	TAB
CLOSE	ERROR	LOC	PRINT	TAN
CLS	EXP	LOCATE	PSET	THEN
COLOR	FIELD	LOF	PRESET	TIME
CONT	FILES	LOG	PUT	TROFF
COPY	FIX	LPOS	READ	TRON
COS	FN	LPRINT	REM	USING
CSAVE	FOR	LSET	RENUM	USR
CSNG	FRE	MAXFILES	RESTORE	VAL
CSRLIN	GET	MERGE	RESUME	VARPTR
CVD	GOSUB	MID\$	RETURN	VDP
CVI	GOTO	MKD\$	RIGHT\$	VPEEK
CVS	HEX\$	MKI\$	RND	VPOKE
		MOD	RSET	WAIT
		MOTOR	RUN	WIDTH
		MKS\$		XOR

Apêndice G

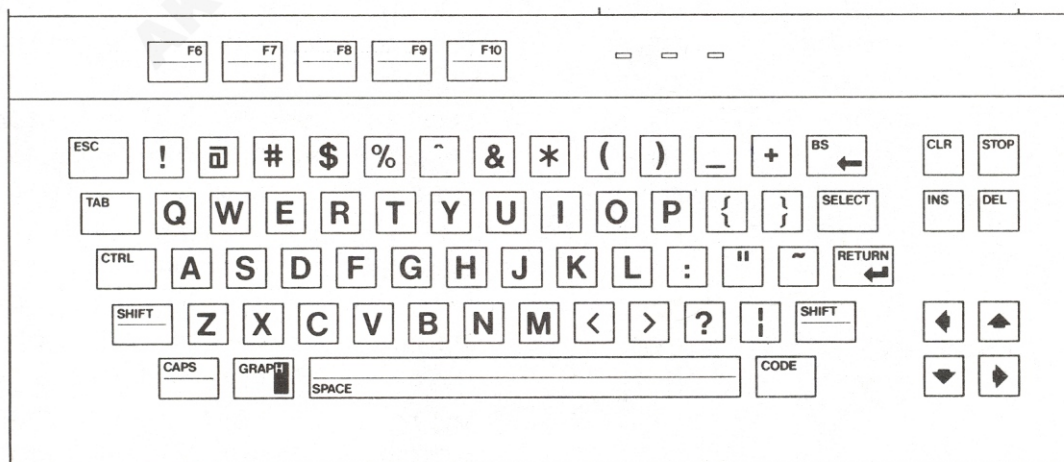
Teclado

Este apêndice dá-nos um resumo completo de todos os caracteres que podem ser produzidos no teclado:

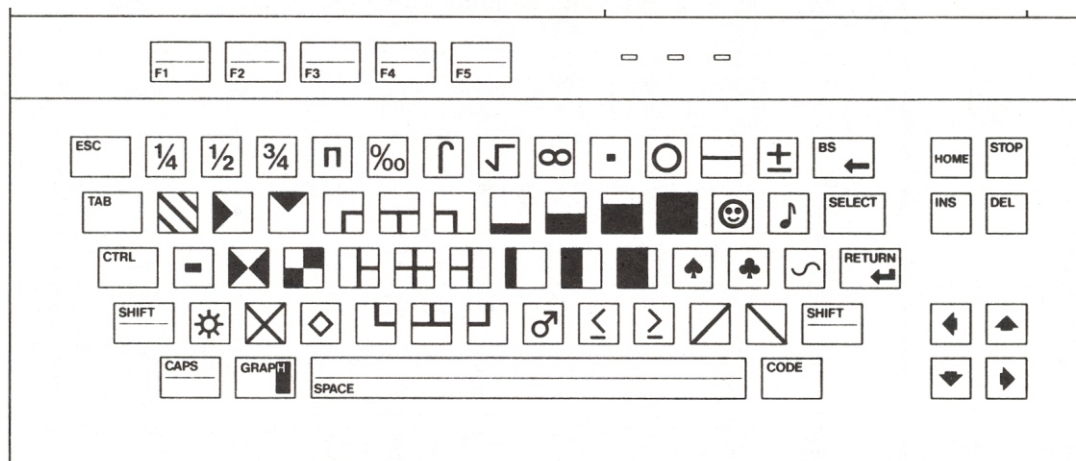
A. Caracteres minúsculos



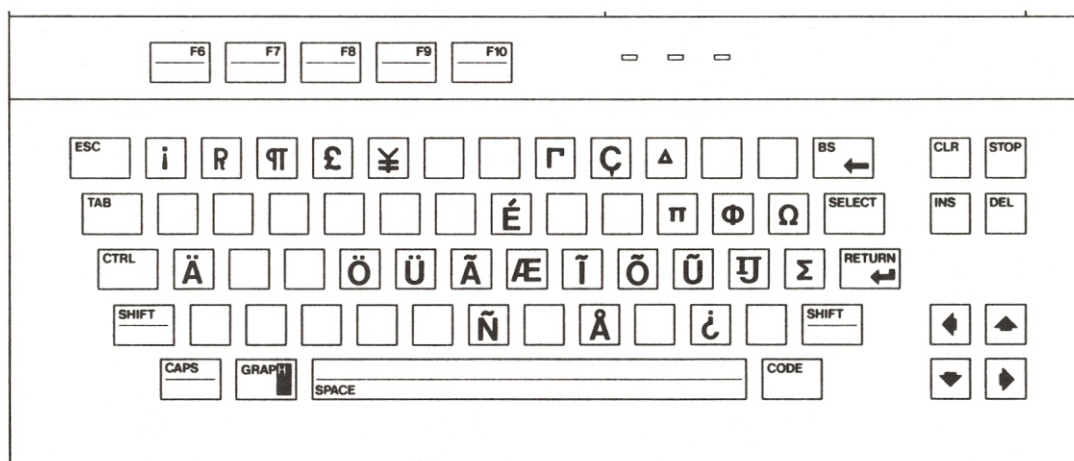
B. Caracteres disponíveis quando a tecla de carácter é premida simultaneamente com a tecla SHIFT



C. Caracteres disponíveis depois da tecla GRAPH ter sido premida



F. Caracteres disponíveis depois de ter sido premida a tecla CODE e quando a tecla do carácter é premiada simultaneamente com a tecla SHIFT



Apêndice H

Especificações técnicas

1. Ordenação do "chip" (circuito integrado)

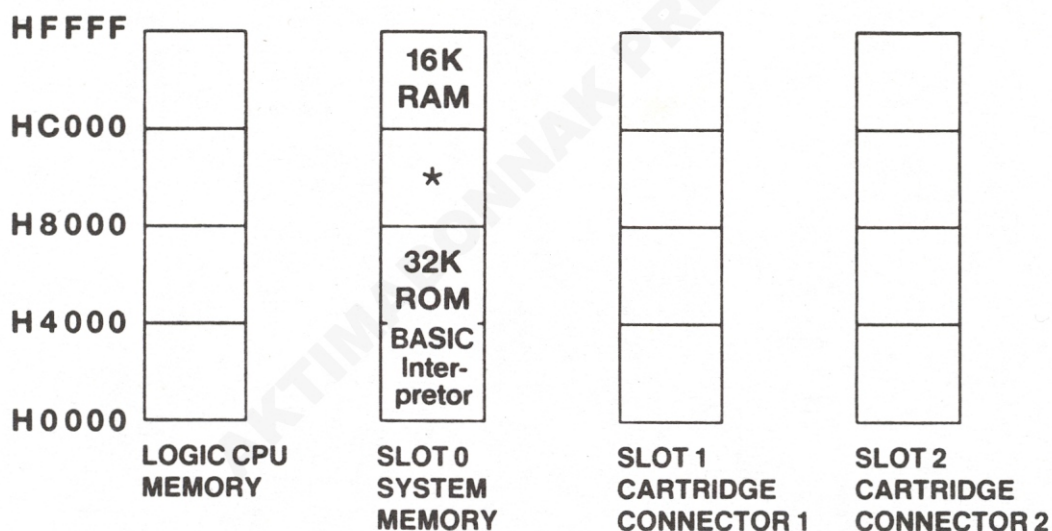
CPU Unidade de processamento central Z80A, 3.5 MHz.

VDP Processador de Video TI TMS-9929A ou circuito integrado similar

PSG Gerador de som programável GI AY-3-8910 ou circuito integrado similar

PPI Ligação programável entre periféricos I 8255

2. Construção da memória



* Esta área é preenchida com outra 16K RAM dentro de 32K RAM MSX-computadores.

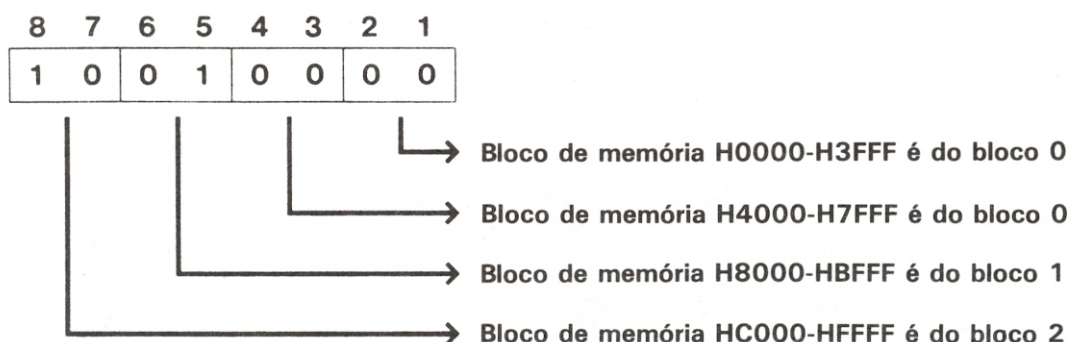
O Computador MSX tem também 16K video RAM (ver apêndice I deste manual).

Com memórias de 16K RAM, os endereços da memória &HC000 a &HC31F e os endereços da memória &HF380 a &HFFFF são usadas pelo computador como áreas de trabalho.

Com memórias de 32K RAM, os endereços da memória &H8000 a &H831F e os endereços da memória &HF380 a &HFFFF são usadas pelo computador como áreas de trabalho.

O espaço lógico da memória cpu é preenchido com espaços físicos da memória ou vários blocos de memória. A maneira como estes blocos da memória são preenchidos é gravada no selector de registo dos blocos da memória, que se encontram na porta A do PPI.

Aqui temos um exemplo:



O endereço da porta A pode ser encontrado dentro de input/output endereço de tabela.

3. Tabela de endereços input/output

HFF		ADRESS	R/W	DESCRIPTION	REMARK
		HA8	W R	Port A data write Port A data read	8255
		HA9	W R	Port B data write Port B data read	PPI
		HAA	W R	Port C data write Port C data read	
HB0	PPI	HAB	W	Mode set	
HA8	PSG				
HA0	VDP	HA0	W	Adress latch	AY-3-8910
H98		HA1	W	Data write	PSG
		HA2	R	Data read	
		H98	W R	V-RAM Data write V-RAM Data read	TMS 9929A
		H99	W R	Command, adress set Status read	VDP
H00					

Com o MSX-BASIC é possível comunicar directamente com as portas input/output utilizando as instruções "OUT" e "WAIT" e a função "INP".

4. Ligações dos módulos e/ou interfaces

Pin 49			1		
					
Pin 50			2		

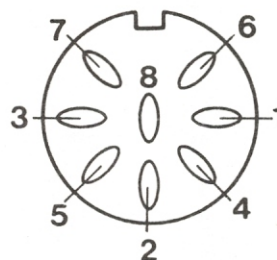
Pin	Name	I/O	Pin	Name	I/O
1	CS1	O	2	CS2	O
3	CS12	O	4	SLTSL	O
5	Reserve	—	6	RFSH	O
7	WAIT	I	8	INT	I
9	M1	O	10	BUSDIR	I
11	IORQ	O	12	MERQ	O
13	WR	O	14	RD	O
15	RESET	O	16	Reserve	—
17	A9	O	18	A15	O
19	A11	O	20	A10	O
21	A7	O	22	A6	O
23	A12	O	24	A8	O
25	A14	O	26	A13	O
27	A1	O	28	A0	O
29	A3	O	30	A2	O
31	A5	O	32	A4	O
33	D1	I/O	34	D0	I/O
35	D3	I/O	36	D2	I/O
37	D5	I/O	38	D4	I/O
39	D7	I/O	40	D6	I/O
41	GND	—	42	CLOCK	O
43	GND	—	44	SW1	—
45	— 5V	—	46	SW2	—
47	— 5V	—	48	+ 12V	—
49	SOUNDIN	I	50	— 12V	—

Pin	Name	Content
1	CS1	ROM addresses 4000 ~ 7FFF select signal
2	CS2	ROM addresses 8000 ~ BFFF select signal
3	CS12	ROM addresses 4000 ~ BFFF select signal (for 256k ROM)
4	SLTSL	Slot select signal
5	Reserve	Reserved signal line ~ use inhibited
6	RFSH	Refresh cycle signal
7	WAIT	CPU's WAIT request signal
8	INT	Interrupt request signal to CPU
9	M1	Signal expressing CPU fetch cycle
10	BUSDIR	This signal controls direction of external data bus buffer Cartridges are selected and L level is output from each cartridge at data transmission time
11	IORQ	I/O request signal
12	MERQ	Memory request signal
13	WR	Write timing signal
14	RD	Read timing signal
15	RESET	System reset signal
16	Reserve	Reserved signal line ~ use inhibited
17 ~ 32	A0 ~ A15	Address bus signals
33 ~ 40	D0 ~ D7	Data bus signals
41	GND	Signal ground
42	CLOCK	CPU clock 3.579545MHz
43	GND	Signal ground
44, 46	SW1, SW2	For insertion/removal protect
45, 47	+ 5V	+ 5V power source
48	+ 12V	+ 12V power source
49	SOUNDIN	Sound input signal (— 5bdm)
50	— 12V	— 12V power source

5. Ligações do gravador

Pin	Name	I/O
-----	------	-----

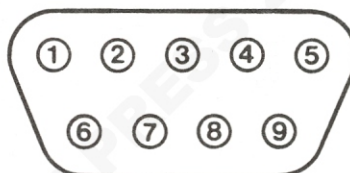
1	GND	
2	GND	
3	GND	
4	CMTOUT	O
5	CMTIN	I
6	REM +	O
7	REM -	O
8	GND	



6. Ligações dos punhos de comando

Pin	Name	I/O
-----	------	-----

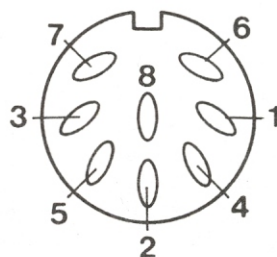
1	FWD	I
2	BACK	I
3	LEFT	I
4	RIGHT	I
5	+ 5 V	
6	TRG 1	I/O
7	TRG 2	O
8	Output	O
9	GND	



7. Ligações do monitor

Pin	Signal PAL version	Signal RGB version*
-----	-----------------------	------------------------

1	+ 5 V	Status RGB
2	GND	GND
3	Sound	Blue
4	Luminance	Luminance
5	Comp. Video	Red
6	+ 12 V	+ 12 V
7		Sound
8		Green



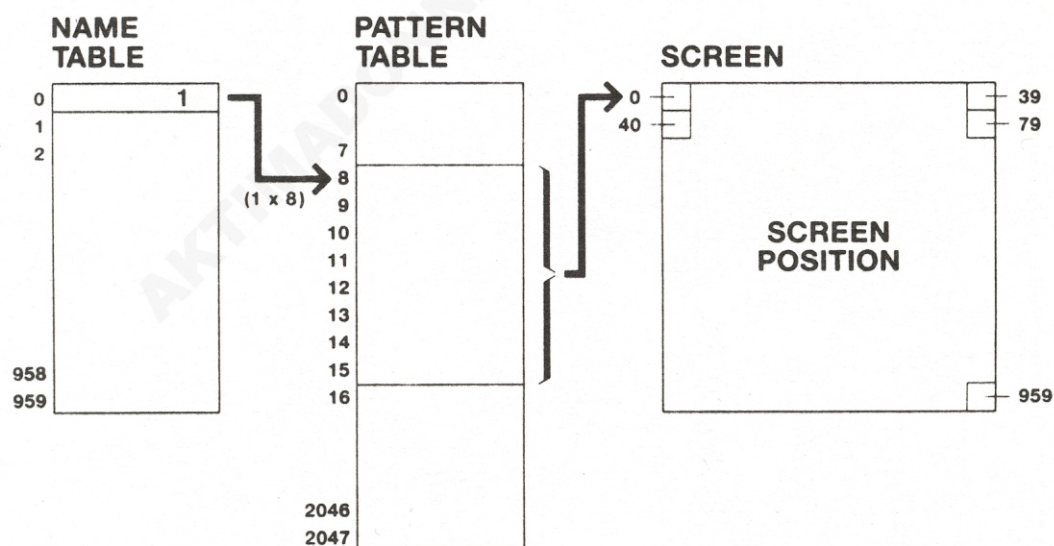
*) RGB in french version only

Apêndice I

Processador de vídeo (VDP)

O computador MSX está equipado com um VDP que fornece todas as informações necessárias para produzir uma imagem no écran. Para o fazer, VDP está por sua vez equipado com uma memória de vídeo de 16K RAM. Podem pedir-se informações do video RAM com a função "VPEEK" e podem ser armazenadas directamente com a instrução "VPOKE". Todas as instruções escritas (impressas) tais como "PRINT" e todas as instruções gráficas tais como "DRAW" são automaticamente processadas pelo MSX-BASIC e colocadas na memória de vídeo RAM. Tudo o que aparece no écran é armazenado na memória de vídeo RAM em tabelas, que são utilizadas de diferentes maneiras, dependendo do modo de écran.

Modo de texto 1 Neste modo o écran está dividido em 24 linhas com 40 posições cada. Com este fim há duas tabelas na memória de vídeo RAM: uma tabela de nomes e uma tabela padrão. A tabela de nomes contém todos os caracteres de código de todos os símbolos que aparecem no écran. Na tabela padrão foram reservados 8 bytes por nome para determinar como é o carácter. A posição na tabela de nomes também indica a posição no écran.



A localização das tabelas na memória Vídeo RAM é indicada pela variável "BASE(n)". "BASE(0)" dá o primeiro endereço da tabela de nomes, enquanto que "BASE(2)" dá o primeiro endereço da tabela padrão.

Exemplo:

```

NEW
10 SCREEN 0:WIDTH 40
20 PRINT "HELLO"
30 PRINT BASE(0),BASE(2)
40 A=VPEEK(BASE(0)):PRINT A
50 B=A*8:FOR I=B TO B+7
60 C$="00000000"+BIN$(VPEEK(BASE(2)+I))
65 PRINT RIGHT$(C$,8)
70 NEXT I
RUN

```

Modo de texto 2

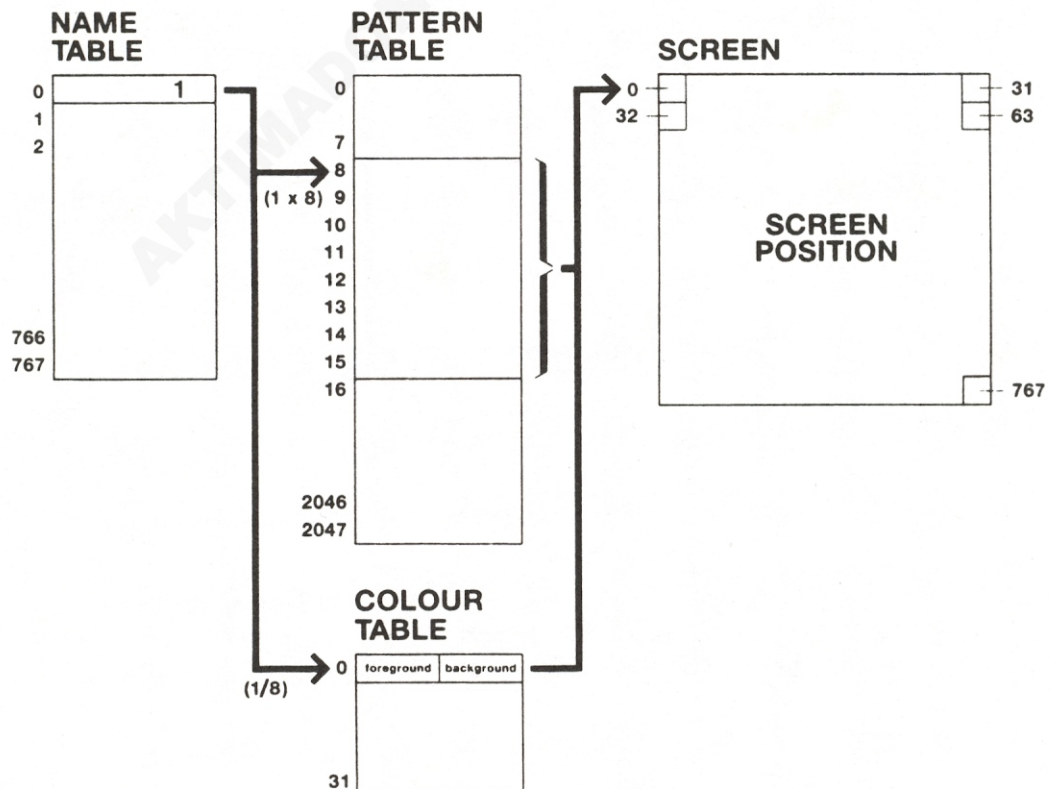
No exemplo acima o primeiro endereço da tabela de nome e da tabela padrão são impressos na linha 30. O código de carácter (nome) da linha "H" é impresso na linha 40. A seguir o padrão da letra "H" é reproduzido de forma binária. Compare estas 8 linhas com o código de carácter da letra "H" no Apêndice E.

Neste modo o écran é dividido em 24 linhas de 32 caracteres cada uma. Com este fim há três tabelas na memória Video RAM: uma tabela de nomes, uma padrão e uma de cores. As tabelas de nomes e a padrão funcionam da mesma maneira que as de modo de texto 1. A tabela de cores contém 32 combinações diferentes de cores de primeiro plano e de cores de fundo.

Caracteres de código 0 a 7 na tabela de nomes utilizam a primeira combinação de cores na tabela de cores. Os 8 códigos seguintes usam a segunda combinação de cores da tabela de cores, etc.

Uma combinação de cores da tabela de cores é armazenada num byte que foi dividido em 2×4 bits. Os primeiros quatro bits indicam a cor de fundo.

A posição da tabela de nomes na memória de Video RAM é registada na variável "BASE(5)", a posição da tabela padrão na variável "BASE(7)" e a posição da tabela de cores na variável "BASE(6)".



Vamos mudar o nosso exemplo anterior como se segue:

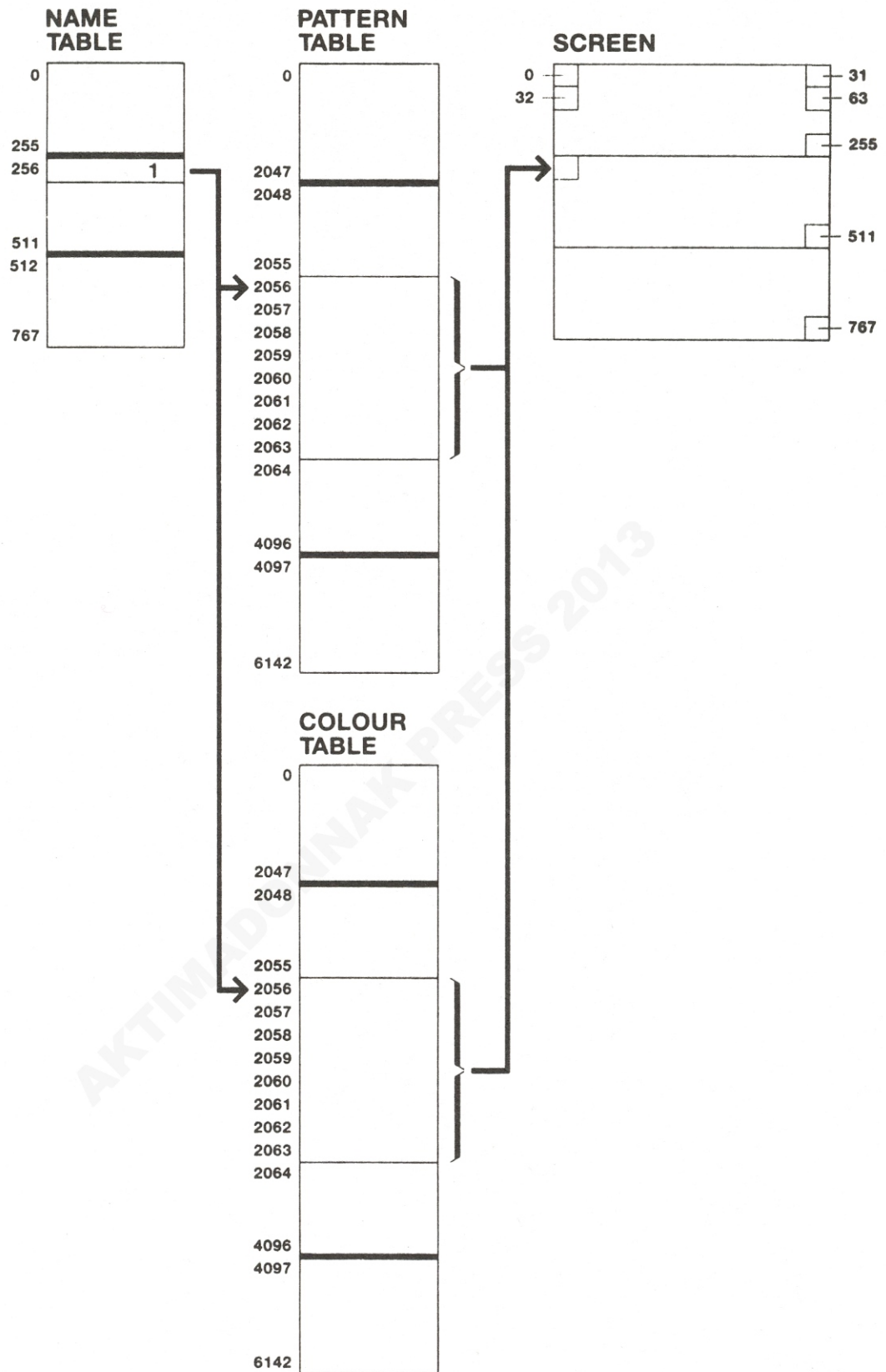
```
10 SCREEN 1:WIDTH 32
30 PRINT BASE(5),BASE(6),BASE(7)
40 A=VPEEK(BASE(5)):PRINT A
60 PRINT C$="00000000"+BIN$(VPEEK(BASE(7)+1))
80 D=VPEEK(BASE(6))
90 PRINT USING "##";D/16
100 PRINT USING "##";D MOD 16
RUN
```

Os primeiros endereços da tabela de nomes, a tabela padrão e a tabela de cores são impressas neste exemplo, seguidos pela forma de carácter e o nome desse carácter. Neste caso a letra "H"

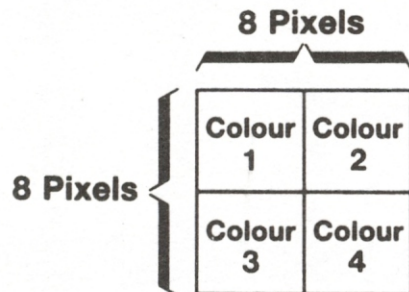
Depois de verem e terem executado este exemplo, escrevam:

```
SCREEN 0:WIDTH 40
```

Modo gráfico 1 Neste modo de écran, este é dividido em 3×8 linhas com 32 posições em cada linha. Este modo de écran tem também um nome de tabela que indica a posição do écran. Um número na tabela de nomes indica onde encontrar o padrão e a cor do carácter a serem impressos. Para cada nome na tabela de nomes encontra-se um padrão que pode ser definido na tabela de padrões. Com este fim a tabela de padrões é de 6144 bytes (768 nomes X 8 bytes). Para cada byte da tabela padrão uma cor de fundo e de primeiro plano podem ser definidas na tabela de cores. É por isso que a tabela de cores tem o mesmo comprimento que a tabela padrão. A posição da tabela de nomes na memória Video RAM está compreendida na variável "BASE(10)" e a posição da tabela de padrão na variável "BASE(12)". A posição da tabela de cores está compreendida na variável "BASE(11)".

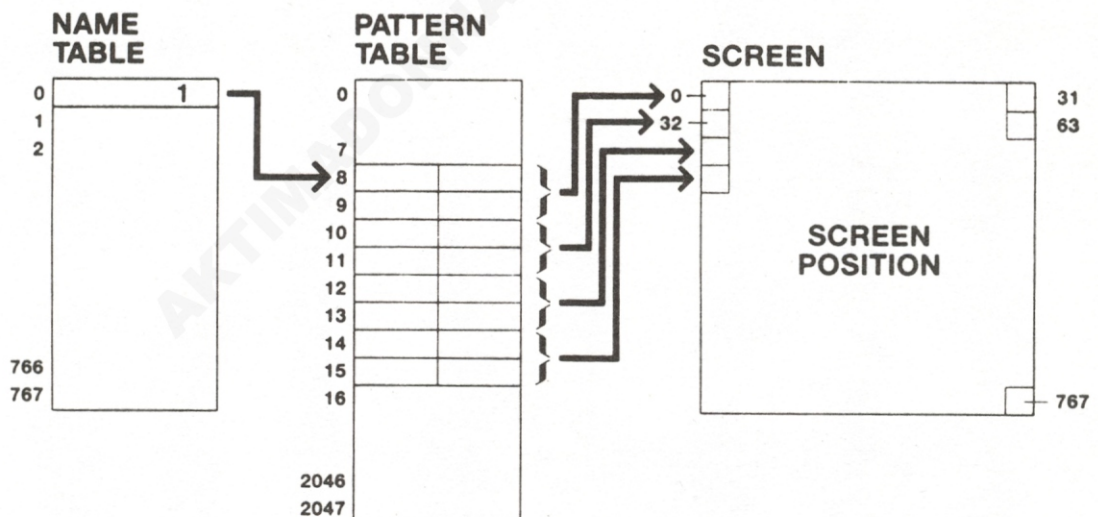


Modo gráfico 2 Neste modo o écran está dividido em 24 linhas com 32 posições cada. Cada posição consiste de quatro quadrados com um tamanho de 4×4 pontos de imagem cada. Cada quadrado contém uma cor. Neste modo de écran encontra-se uma tabela de nomes que também indica a posição no écran. O número na tabela de nomes indica onde se podem encontrar as cores da posição no écran da tabela padrão.



Cada byte na tabela padrão foi dividido em 2×4 bits e cada combinação de 4 bit indica um número de cor. Cada nome na tabela de nomes requer 3 bytes na tabela padrão. Apenas 2 bytes do segmento de 8 byte são utilizados para especificar a imagem no écran. Os primeiros 2 bytes nos grupos de segmentos de 8 byte são usados para os nomes da fila de cima. A fila seguinte utiliza os bytes 3 e 4 nos segmentos de 8 byte. A fila seguinte utiliza os bytes 5 e 6 enquanto a última fila utiliza os bytes 7 e 8. Esta série é repetida para o resto do écran.

A posição da tabela de nomes na memória RAM de video está compreendida nas variáveis "BASE(15)". A posição da tabela padrão está compreendida na variável "BASE(17)".



"Sprites"
(imagens
que passam
rapidamente
no écran)

Todos os "sprites" são também colocados na memória RAM de video. A definição do "sprite" está resgistada na tabela padrão de "sprites", enquanto que a posição do "sprite" no écran está registada na tabela de atributo. A tabela padrão de "sprites" consiste em 8 bytes para pequenos "sprites" e 32 bytes para "sprites" grandes. Um máximo de 32 posições de "sprites" pode ser indicada na tabela de "sprites" atribuída.

Cada entrada de "sprite" na tabela de atributo de "sprites" consiste em 4 bytes:

	7	6	5	4	3	2	1	0	Bit
Byte 0	Y-position								
1	X-position								
2	Sprite name								
3									Colour

Um valor pode ser colocado na posição byte Y, de 0 a 255. Este número determina a linha no écran, fornecendo a seguinte informação:

0-191 é a linha no écran.

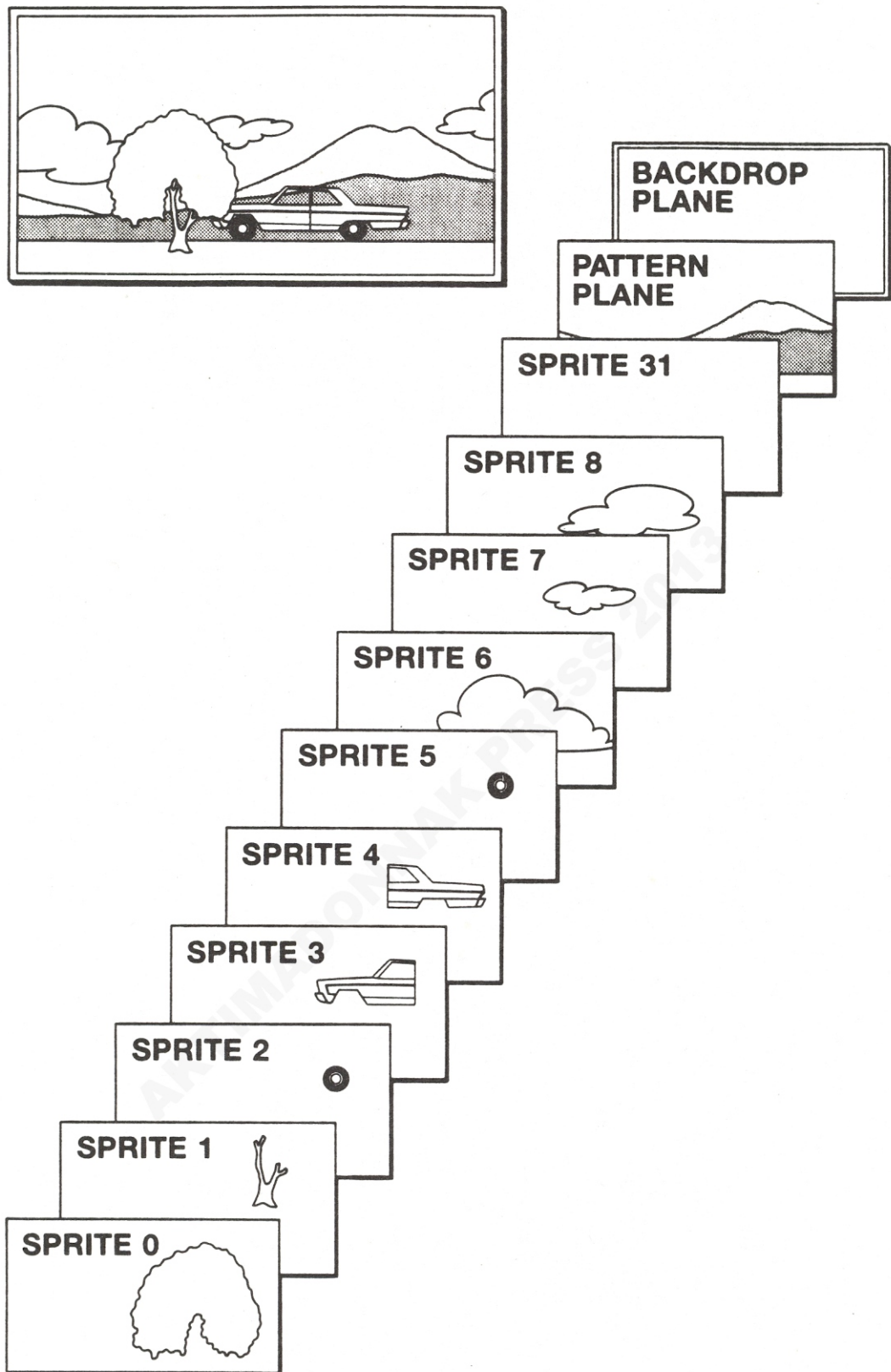
208 significa que o "sprite" já não é projectado no écran.

224-255 representa as linhas no écran - 31 a - 1. Isto torna possível colocar um "sprite" acima da linha superior.

Um valor de 0 a 255 pode ser distribuído para a posição X do byte. Este número determina a coluna no écran. Se o bit 7 da cor byte foi igual a 1, então a posição da coluna do "sprite" é igual ao valor da posição X menos 32. Se a posição X for igual a 0 e acontece que o bit 7 da cor byte 1, então o "sprite" será colocado na coluna - 32. Isto torna possível que um "sprite" seja colocado antes da primeira coluna.

As posições Y e X dadas aplicam-se à imagem superior esquerda do "sprite".

O "sprite" que foi colocado em primeiro lugar na tabela de "sprites" tem a mais alta prioridade (sprite 0).



Na figura da página anterior vê-se um carro numa paisagem:

A árvore é feita dos "sprites" 0 e 1.

O carro dos "sprites" 2 a 5.

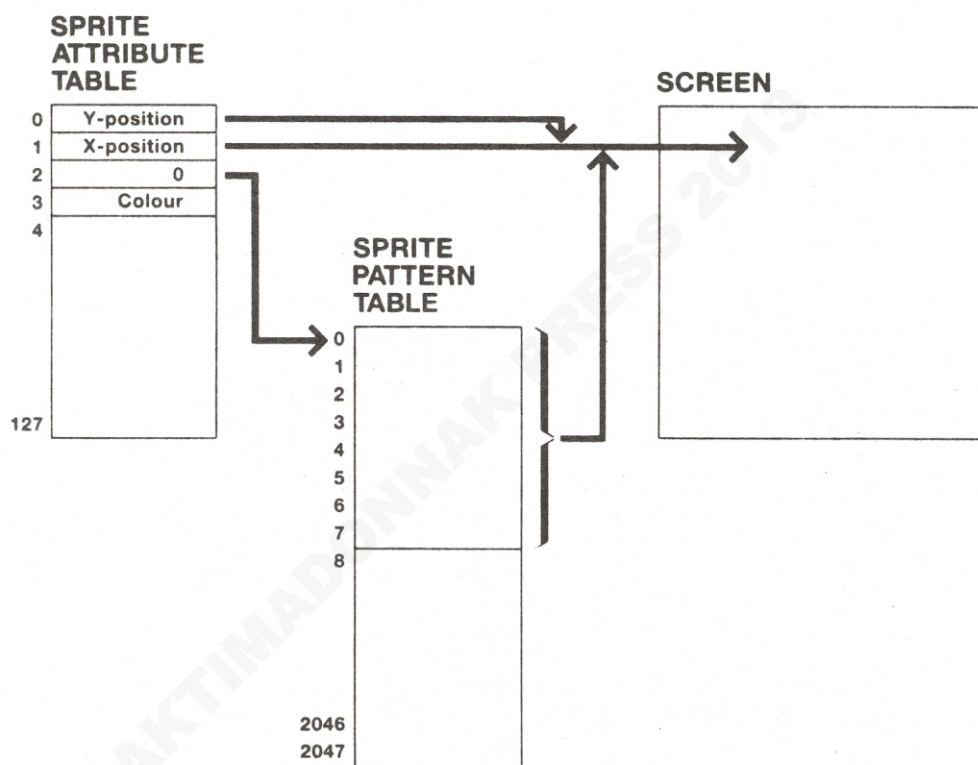
As nuvens dos "sprites" 6, 7 e 8.

Uma vez que os "sprites" 0 e 1 têm prioridade sobre os "sprites" 2 e 5, a árvore aparece em frente do carro.

O nome do "sprite" da tabela de "sprites" é um número que indica onde se pode encontrar o padrão de "sprite" na tabela padrão dos "sprites".

O primeiro endereço da tabela de "sprites" na memória RAM de video está contido na variável "BASE(8)" no modelo de texto 2, em "BASE(13)" no modo gráfico 1 e na "BASE(18)" no modo gráfico 2.

O primeiro endereço da tabela padrão de "sprites" está contido na variável "BASE(9)" no modo de texto 2, na "BASE(14)" no modo gráfico 1 e na "BASE(19)" no modo gráfico 2.



Limpe-se a memória com o comando "NEW" e tente-se o exemplo seguinte:

```
NEW
10 SCREEN 1:WIDTH 29
20 DATA 16,56,68,254,16,16,16,16
30 FOR I=0 TO 7
40 READ A:VPOKE(BASE(9)+I),A
50 NEXT I
60 PRINT:PRINT:PRINT
70 PRINT "A SPRITE IN TEXT MODE 2"
80 VPOKE (BASE(8)+0),40
90 VPOKE (BASE(8)+2),0
100 VPOKE (BASE(8)+3),1
110 FOR I=0 TO 255
```

```

120 VPOKE (BASE(8)+1),I
130 NEXT I
140 GOTO 110
RUN

```

Este exemplo mostra que é possível, apesar de tudo, usar os "sprites" no modo do texto 2 comunicando directamente com a memória RAM VDP. Premir simultaneamente CTRL e STOP para parar o programa.

A seguir escreva-se:

```
SCREEN 0:WIDTH 40
```

Registos O VDP tem 9 registos disponíveis: os registos de 0 a 7 e o registo 8. Com o MSX-BASIC o conteúdo dos registos está contido nas variáveis "VDP(n)", sendo "n" um número entre 0 e 8. Com excepção para o registo 8, estas variáveis podem ser alteradas. O significado dos registos é o seguinte:

	0	1	2	3	4	5	6	7	Bit
Register 0	0	0	0	0	0	0	A	D	
Register 1	1	0	E	B	C	R	S	M	
Register 2	0	0	0	0	Name table				
Register 3	Colour table								
Register 4	0	0	0	0	0	Pattern table			
Register 5	0	Sprite attribute table							
Register 6	0	0	0	0	0	Sprite pattern table			
Register 7	Text colour 1				Text colour 2				
Register 8	T	N	V	5 th Sprite					

ABC

000 = Modo texto 2

100 = Modo gráfico 1

001 = Modo gráfico 2

010 = Modo texto 1

D = 0 significa que não há VDP input externo

D = 1 significa que há input externo

E = 0 significa que não há interrupção VDP

E = 1 significa que há interrupção VDP

R foi reservado

S = 0 significa que os "sprites" são 8×8

S = 1 significa que os "sprites" são 16×16

M = 0 significa que os "sprites" não são aumentados

M = 1 significa que os "sprites" são aumentados

Tabela de nomes = primeiro endereço na tabela de nomes na memória RAM do VDP

Tabela de cores = primeiro endereço na tabela de cores na memória RAM do VDP

Tabela padrão = primeiro endereço da tabela padrão na memória RAM do VDP

Tabela de "sprites" = primeiro endereço na tabela atribuída aos sprites na memória RAM VDP

Tabela padrão de "sprites" = primeiro endereço na tabela padrão de "sprites" na memória RAM VDP

Cor 1 do texto = Cor de primeiro plano do modo de texto 1

Cor 2 do texto = Cor de fundo do modo de texto 1

T O T é o marcador do registo 8 quando levado a 1 no painel do rastreio da última linha de varrimento do écran. É levado a 0 depois do registo ser lido ou do VDP ser activado exteriormente para "RESET".

N O marcador N do registo 9 é levado a 1 se dois ou mais sprites coincidem. A coincidência ocorre se dois "sprites" no écran tiverem um ponto sobreposto (pixel). O marcador N é apagado para 0 depois do registo 8 ser lido ou o VDP ser activado exteriormente a RESET.

O registo 8 deve ser imediatamente lido após ser ligado a alimentação para assegurar o marcador a 0 "RESET".

O VDP testa cada posição de "pixel" para coincidência durante a produção do "pixel" em relação onde está calculado no écran. Isto dá-se durante 1/50 de segundo no TMS9929A. Então, quando movendo "sprites" mais que uma posição de "pixel" na duração destes intervalos é possível que "sprites" tenham múltiplos "pixels" sobrepostos até que tenham passado uns pelos outros quando o VDP verificar a coincidência.

V e o 5º "sprite" O marcador V no registo 8 é levado a 1 sempre que haja cinco ou mais "sprites" numa linha horizontal (linhas de 0 a 192). O marcador V é apagado para 0 depois do registo 8 ser lido ou o VDP ser activado em RESET. O número do 5º "sprite" é colocado nos 5 bits inferiores do registo 8 quando o marcador V é indicado e é válido sempre que o marcador V é 1.

As tabelas na memória RAM de video podem ser colocadas pelo utilizador numa posição diferente no programa desde que cumpra as seguintes regras MSX-BASIC:

1. A tabela de nomes deve começar no endereço 0 ou num endereço indicado por um múltiplo de 1024.
2. A tabela de cores deve começar no endereço 0 ou num endereço indicado por um múltiplo de 64.
3. A tabela de referência deve começar no endereço 0 ou num endereço indicado por um múltiplo de 2048.
4. A tabela atribuída aos "sprites" deve começar no endereço 0 ou num endereço indicado por um múltiplo de 128.
5. A tabela padrão dos "sprites" deve começar no endereço 0 ou num endereço que é indicado por um múltiplo de 2048.

AKTIMADONNAK PRESS 2013

Apêndice J

Gerador de som programável

O computador MSX está preparado com um gerador de som programável (PSG) que produz o som. Com este fim o PSG tem 16 registos, a que se pode dar um valor directamente com a instrução "SOUND" do MSX-BASIC. Em contraste com a instrução "PLAY", o som fica até que o registo a que pertence volte a 0 com outra instrução "SOUND".

Os registos têm os seguintes significados:

	7	6	5	4	3	2	1	0	Bit
Register 0	Tone number channel A								
Register 1	0	0	0	0	Coarse tone A				
Register 2	Tone number B								
Register 3	0	0	0	0	Coarse tone B				
Register 4	Tone number channel C								
Register 5	0	0	0	0	Coarse tone C				
Register 6	0	0	0	Noise tone					
Register 7	0	0	RC	RB	RA	TC	TB	TA	
Register 8	0	0	0	M	Volume A				
Register 9	0	0	0	M	Volume B				

Register 10	0	0	0	M	Volume C
Register 11					Form-length tone
Register 12					Form-length coarse tone
Register 13	0	0	0	0	Form number

O tom para os canais A, B e C é colocado nos registros de 0 a 5. O registro 7 determina para que canal o tom indicado ou ruído, produzido pelo circuito integrado (chip), é dirigido. Se o registro 7 tem um conteúdo binário de 00111110 significa que uma tonalidade é apenas reproduzida através do canal A.

O volume para todos os canais é colocado nos registros 8 a 10. Quando um bit M é colocado em 1 significa o volume máximo.

Os registros 11 e 12 determinam o comprimento da forma escolhida no registro 13.

O número da forma no registro 13 tem o mesmo significado que as sub-comandos na instrução "PLAY".

O registro 14 é usado como porta A de input/output e o registro 15 como porta B de input/output.

Número do tom O tom para cada canal é determinado como segue:

$$TP = \frac{CF}{16 \times TF} \quad CT + \frac{T}{256} = \frac{TP}{256}$$

TP = Período do tom

CF = Frequência de relógio (= 1.78977 MHz)

TF = Frequência do tom

T = Tom (registros 0, 2, 4)

CT = Tom harmónico (registros 1, 3 5)

Exemplo:

Um tom de 440 Hz do canal A é determinado como segue:

$$TP = \frac{1.78977 \times 10^6}{16 \times 440} = 254 \quad CT + \frac{T}{256} = \frac{254}{256}$$

$$CT = 0 \quad T = 254$$

Registro 0 = 254 e registro 1 = 0

Acompanhamento O acompanhamento é determinado como segue:

$$NP = \frac{CF}{16 \times NF} \quad NT = NP$$

NP = Período de acompanhamento

CF = Frequência de relógio (= 1.78977 MHz)

NF = Frequência do acompanhamento

NT = Tom do acompanhamento (registro 6)

Exemplo:

O tom de acompanhamento de 30 kHz, por exemplo, é determinado como segue:

$$NP = \frac{1.78977 \times 10^6}{16 \times (30 \times 10^3)} = 4 \quad NT = 4$$

Registro 5 = 4

Extensão do tom A extensão do som é determinada da seguinte maneira:

$$SP = \frac{CF}{256 \times SF} \quad CT + \frac{T}{256} = \frac{SP}{256}$$

SP = Extensão do período

CF = Frequência de relógio (= 1.78977 MHz)

SF = Extensão de frequência

T = Forma de extensão do tom (registro 11)

CT = Forma da extensão do tom e dos sons harmónicos (registro 12)

Exemplo:

A forma de frequência de 0.5 Hz é assim determinada:

$$SP = \frac{1.78977 \times 10^6}{256 \times 0.5} = 14000 \quad CT + \frac{T}{256} = \frac{14000}{256}$$

$$CT = 54 \quad T = 176$$

Registro 11 = 176 e registro 12 = 54

Apagar a memória com o comando "NEW" e escrever o exemplo seguinte:

```
10 FOR I=0 TO 13
20 SOUND I,0
30 NEXT I
40 SOUND 7,&H3E
50 SOUND 8,&H0F
60 FOR I=&H10 TO &HFO
70 SOUND 0,I
80 NEXT I
90 SOUND 6,&H0F
100 SOUND 7,&H07
110 SOUND 8,&H10
120 SOUND 9,&H10
130 SOUND 10,&H10
140 SOUND 12,&H40
150 SOUND 13,&H00
RUN
```

Nas linha 10 a 30 todos os registos são postos a zero num ciclo "FOR...NEXT".
Na linha 40 apenas um tom do canal A é permitido. Na linha 50 é determinado o volume do canal A.

Nos ciclos "FOR...NEXT" das linhas 60 a 80, tons sucessivos são colocados para o canal A.

Na linha 90 os harmónicos são determinados.

Na linha 100 é indicado que os harmónicos devem ser reproduzidos nos canais A, B e C.

Nas linhas 110 a 130 é dado o volume máximo para os três canais.

Na linha 140 a extensão da forma é determinada.

Na linha 150 é determinada a forma dos harmónicos.

o Mar O exemplo seguinte dá uma imitação do som do mar:

```
NEW
10 FOR I=0 TO 13
20 SOUND I,0
30 NEXT I
40 SOUND 6,&H10
50 SOUND 7,&H07
60 SOUND 8,&H10
70 SOUND 9,&H10
80 SOUND 10,&H10
90 SOUND 12,&H40
100 SOUND 13,&H0E
RUN
```

Todos os sons que ouvimos são produzidos no circuito integrado de som. Depois de termos executado o programa, o MSX-BASIC fica ao nível de comando enquanto que o som continua.

Apenas parará quando os registos 6 e 13 chegarem a 0 ou se se premirem simultaneamente as teclas CTRL e STOP.

AKTIMADONNAK PRESS 2013



PHILIPS

PHILIPS
LUMINEA

